MACHINE INTELLIGENCE FOR IOT AND EDGE

D3.2 Report on Data Sharing Policies & Processes between Edge Devices

Work Package	WP3
Dissemination level	Public
Status	Final
Date	6/7/2023
Deliverable leader	Mário Sousa, FEUP
Potential Contributors	All

Contributors

Names	**Organisation**
Sena Çağlar	Enforma
Barış Bulut	Enforma
Sencer Sultanoglu	Eliar
Mustafa Çom	Eliar
Anna Hristoskova (AH)	Sirris
Nicolás González-Deleito (NGD)	Sirris
Sarah Klein	Sirris
Gregoire de Hemptinne	Shayp
Mojtaba Eliassi	3E
Julien Deckx	3E
Geert Vanstraelen	Macq
Pedro Santos (PS)	ISEP
Mario Sousa	FEUP
Luís Almeida (LDA)	FEUP
Joana Sousa (JS)	NOS
João Tagaio	NOS
Nuno Martins	NOS

Reviewers

Names	Organisation
Sreeraj Rajendran	Sirris
Annada Rath	Sirris

Rev. #	Responsible	Comments	Date
0.1	PS	Deliverable: structure and intro.	
0.2	PS		
0.3	PS		
0.6	PS	Contributions by 3E	
0.7	PS	Final edits and wrapping up before revision	2023/04/04
1.0	LDA	Final edits after internal revision	2023/06/22

Table of Contents

List of Figures

Abbreviations

AI	Artificial Intelligence
ANPR	Automatic Number Plate Recognition
API	Application Programming Interface
AWS	Amazon Web Services
CPE	Customer Premises Equipment
DDoS	Distributed Denial-of-Service
EARS	Easy Approach to Requirements Syntax
GCP	Google Cloud Platform
HGW(s)	Home Gateway(s)
IoT	Internet of Things
IR	Infrared
MFBB	MIRAI Framework Building Blocks
ML	Machine Learning
MW	Megawatt
MWh	Megawatt-hour
NB-IoT	Narrowband Internet of Things
NN	Neural Network
OPC	Open Platform Communications
PID	Proportional-Integral-Derivative
PLC	Programmable Logic Controller
PV	Photovoltaic
UC(s)	Use Case(s)
VRU	Vulnerable Road User

Executive Summary

This document provides the deliverable of T3.2 of the ITEA3-MIRAI project and contains the data sharing policies and processes between edge devices for the MIRAI Framework. This task will focus on the mechanisms needed in the framework and on the processes through which data is to be shared and distributed in the MFBB.

Towards producing this output, the following activities of the requirement elicitation phase were carried throughout T3.2:

(1) define data sharing policies: we envision to focus on the design of a secure, reliable, and risk-constrained context-aware access control model that can be used to effectively control and manage data access and data sharing between edge nodes, edge devices and the cloud. The resulting policy model will be designed in such a way so that it can express fine-grained and complex access control policies with high degree of flexibility, adaptable to the different applications envisioned in MIRAI.
(2) policies enforcement and management: this task focuses also on the design of secure (T3.3) and effective policy enforcement mechanisms. This mechanism will ensure that all entities defined in the policy can only perform actions on data/resources allowed to them and all the mishandling of data can be detected, prevented and reported.
This task will take input from WP1 and related tasks T3.1 and T3.3 to develop policies for different types of data and processes for how to monitor and govern data sharing in the platform. Found commonalities will be leveraged to develop the MFBB in a way that may flexibly serve a variety of end users.

This document contains the following elements produced throughout the task: the data streams and processes of the infrastructure of each partner (Chapter 2); data sharing considerations or restrictions in each UC (Chapter 3.1); a framework used to describe the data sharing policies (Chapter 4); and the data sharing policies to be applied in each UC (Chapter 5).

1 Introduction

We start this chapter by introducing terminology relevant to the comprehension of the document. The procedure followed to produce the list of data sharing policies between edge nodes, as well as other relevant information, is presented next. The structure of the remainder of the document is also described.

1.1 Relevant Terminology

The definition of some terms is laid out for the sake of keeping the discussion throughout this document clear and unambiguous.

- **MIRAI Framework (MF, or simply "Framework"):** the Edge AI architecture being proposed under MIRAI. As stated in the **Rationale of the project** section in the FPP (or subsequent iterations): "MIRAI's mission is to design and create the first truly scalable edge computing software toolkit (MFBB) tailored for IoT and edge computing applications."
- **MIRAI Framework Building Blocks (MFBB):** components providing functionalities of the MIRAI Framework. It can refer to both conceptual specifications of MFBBs, or concrete instantiations of MFBBs in the target devices.
- **Target process/equipment:** process and/or equipment being monitored and/or controlled in each use-case (UC).
- **Business service:** the service that the UC owner currently provides and is planned for improvement by application of the MIRAI Framework.
- **Infrastructure:** the set of existing components (hardware or software; computing or communications-oriented; etc.) and any other kind of infrastructure that provide the business service. Typical key physical components are **cloud, edge nodes** and **end nodes**. The first two follow common definitions. End nodes are highly resource-constrained embedded devices that can be the sources or sinks of data but do not possess the computing or storage capabilities to carry out edge computing, particularly running MFBBs. Use cases may or may not include end nodes.
- **MIRAI Service (or "Service"):** the service that the MIRAI framework is providing in each particular UC, improving the existing business service.
- **Solution:** the solution to be developed under the MIRAI project and that will provide the MIRAI service. Solutions will be composed primarily of MFBBs but builds on, and/or interfaces with, and/or integrates, components of the existing infrastructure that provide the business service. A solution can be thought of as a concrete instantiation of the MIRAI Framework. The term "solution" may refer specifically to MFBBs on the edge node, on the cloud nodes, or to all modules regardless of location (unless explicitly stated, the particular case must be inferred from context).
- **System operator:** person or agent in charge of managing the complete system, i.e., the set of the existing target process/equipment and infrastructure, and the solution provided by the MFBBs.

1.2 Use-cases Driving the MIRAI Framework

The following use-cases motivate and drive the development of the MIRAI framework.

- Use case 1: Distributed renewable energy systems (UC owner: 3E)
- Use case 2: Secure Internet provisioning (UC owner: NOS)
- Use case 3: Traffic management (UC owner: Macq)
- Use case 4: Water management (UC owner: Shayp)
- Use case 5: Continuous auto configuration of industrial controllers at edge (UC owner: Eliar & Enforma)

1.3 Procedure to Identify Necessary Data Sharing Policies

We describe the procedure to elicit the data sharing needs and policies for the MIRAI use-cases. It should be noted that, ultimately, the data sharing policies will be described in the format of a Generic Data Sharing Policies Description Framework, explained in Chapter 4.

1) We start by analysing UCs from a data perspective; this is described in Chapter 2.
    a) Define system architecture and components for each UC.
    b) Identify types of data and dataflows between system components.
    c) Identify ownership of data instances and system components.
2) Taking the elements above, we identify where data sharing policies are necessary. The output of this step is reported in Chapter 3.1.
3) Using the Generic Data Sharing Policies Description Framework, map the data sharing policies identified in the previous point into the proposed framework. This is presented in Chapter 5.

Given the variety of infrastructures throughout the various UCs, it is left to each UC partners to implement the produced policies in their infrastructure.

1.4 Document Structure

The remainder of this document is organized as follows.

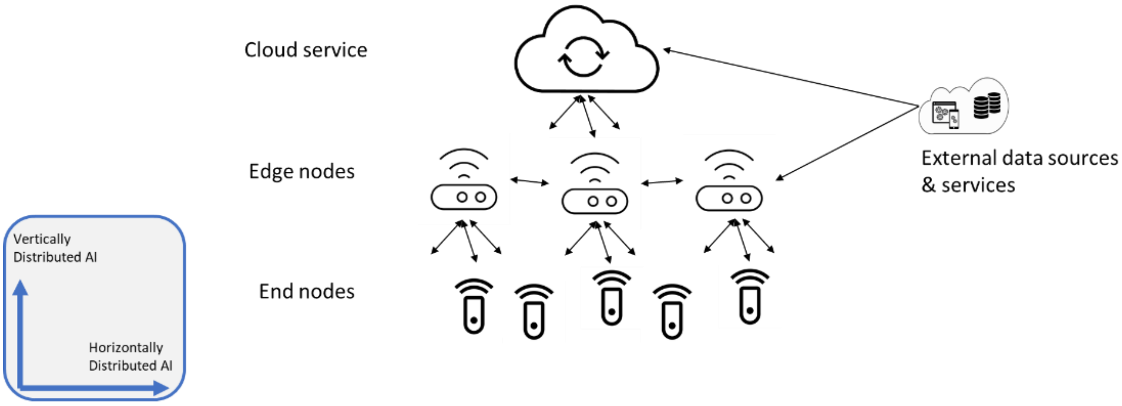
- Chapter 2 presents the data streams and processes of the infrastructure of each partner.
- Chapter 3.1 lists the data sharing considerations or restrictions in each UC.
- Chapter 4 defines the framework used to describe the data sharing policies.
- Chapter 5 presents the data sharing policies to be applied in each UC.
- Chapter 6 draws some final remarks on this deliverable.

2 UC Data Streams and Processes

2.1 High-level MIRAI Framework Architecture

In the MIRAI architecture we distinguish three kinds of nodes: Central or Cloud, Edge and End nodes.

This supports distribution of AI both vertically and horizontally. By 'vertically' we mean from End to Edge and Cloud. 'Horizontal' means among Edge nodes.



MIRAI applications (or *solution*) follows a Service Oriented Architecture, consisting of multiple Building Blocks exchanging data between themselves. The Building Blocks may be running on the Edge nodes or on the cloud, therefore data may be exchanged inside an edge node, or need to travel over a network when the Building Blocks exchanging data are on distinct Edge devices (horizontal distribution), in the cloud, or on distinct layers (e.g. Vertical distribution between Edge nodes and Cloud).

2.2 Use case 1: Distributed renewable energy systems (UC owner: 3E)

The prosumer (producer and consumer) in a commercial and industrial (C&I) prosumer use case typically generates power using a photovoltaic (PV) system and stores it in a battery for later usage. To maximize cost savings and energy efficiency, managing the PV and storage systems might be difficult. Unpredictability in power generation and power consumption, along with inaccuracy of mathematical models of assets lead to the under exploitation of resources. Joint edge cloud functionalities are scoped in MIRAI for this use case to alleviate this issue.

Joint edge cloud functions in this case refer to a vertically distributed computing architecture that combines edge computing (performed at the edge of the network on the CloudGate device, close to the source of data) and cloud computing (performed on remote servers of 3E's SynaptiQ accessed over the internet) to optimize the management of PV and storage systems in real-time.

Most of the MIRAI's applications for this use case such as compression, short term PV and load forecasting, and re-optimization blocks run inside this edge device. The remaining blocks all run in a cloud backend controlled by 3E. The joint edge-cloud system can maximize cost and energy savings by managing the PV and storage systems in a way that combines short-term and real-time data processing at the edge of the system with the long-term cloud-based data analysis and modelling.

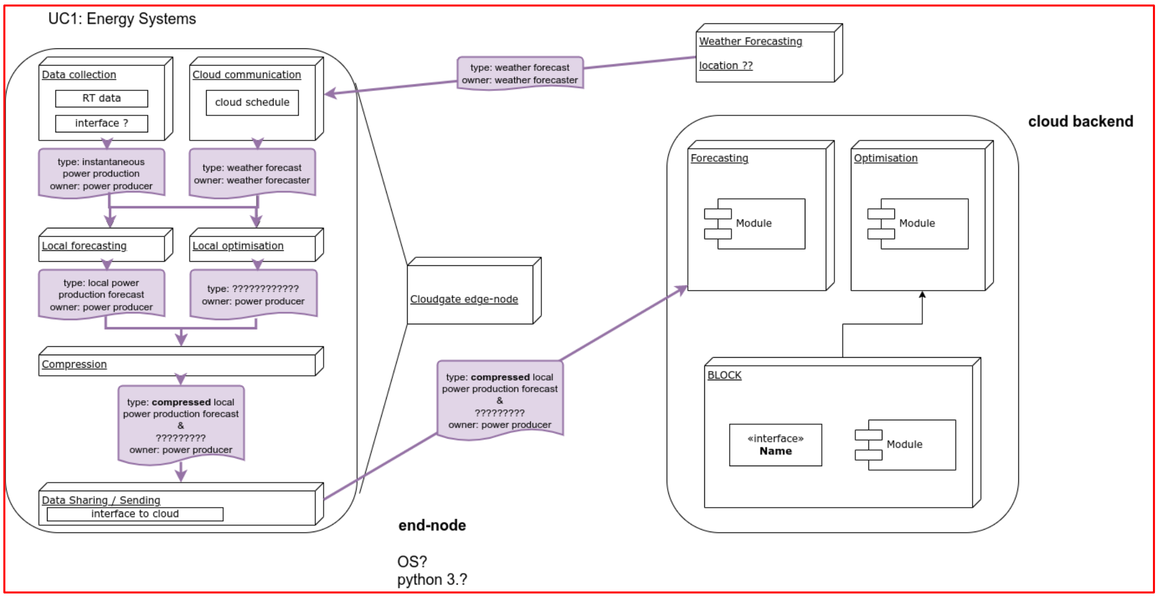


Figure 2-1-Component Diagram and Data Flows (UC 1)

The system cloud computing component offers extra processing power and storage space for doing more intricate data analysis and modeling. Longer-term decisions about how to optimize the PV and storage systems are made using past data on weather, battery capacity, and energy consumption trends. Predicting energy consumption, simulating various storage scenarios and improving the system performance over time are all part of this.

The system edge computing component gathers information on the weather, energy use trends and battery capacity in real-time from the local assets and day-ahead forecasts and schedules from the cloud. In order to decide how much energy to utilize from the PV system and how much energy to store in the battery, the edge computing component subsequently conducts short-term forecasting of PV and consumption together with re-optimization of the day-ahead battery storage schedules. An exemplary modification of a calculated battery schedule in the cloud is shown in the figure below by the re-optimization running on the edge.

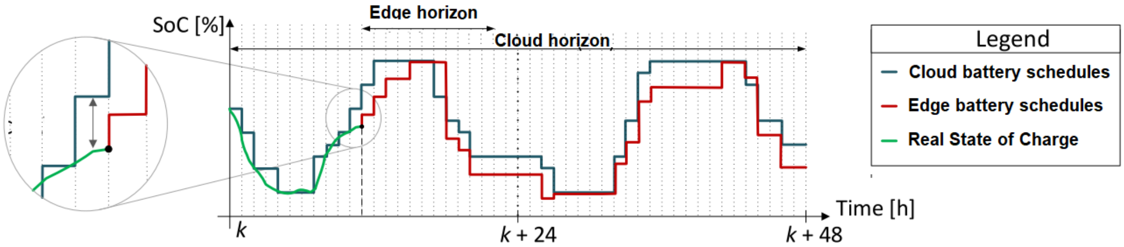


Figure 2-2-Re-optimization process of battery schedules on the edge with a reduced horizon

Block	Description
Local Forecasting	Create forecast for future local power production and consumption, based on current instantaneous power production, historical local production, and forecasts obtained from cloud
Local Optimization	Using short-term forecast to modify the battery schedules received for cloud
Compression	Compress local measured data to be exchanged with the cloud server
Data Sharing	Send locally calculated data to cloud

Cloud communication	Collect day-ahead 15-min weather, forecasts and schedules from cloud
Local Data Collection	Collect data from local assets including power production inverters, load consumptions, storage units (Instantaneous power production)

Dataflows[1]	Description
Hourly local assets (PV, storage, grid meter, consumption, meteorological) data	Via either an FTP push or an API interface, SynaptiQ cloud gathers the site measured data. In essence, SynaptiQ cloud can store per minute pushed data, but in most of use cases, a 15-min granularity suffices. Data may have privacy issues (GDPR compliance).
Real-time (per 5 second) local assets monitoring on the cloud	3E SynaptiQ uses CloudGate on edge for the remote monitoring and control of assets via a dedicated dashboard. In this case, CloudGate provides protocol drivers to communicate with all the local assets. Data may have privacy issues (GDPR compliance).
3rd party meteorological data to 3E Cloud	Added to the locally measured meteorological data, SynaptiQ retrieves predicted and historical meteorological data such as satellite-based irradiance and temperature.
Day-ahead forecasts and battery schedules	By means of forecasting and optimization engines on cloud, SynaptiQ predicts the next-day forecasts of PV and consumption, twice per day. Using these forecasts, the optimization engine determines battery charge/discharge commands each hour as per 15 minutes. Both forecasts and schedules together with the meteorological data are sent every hour to the edge device with 15-min granularity. Data may have privacy issues (GDPR compliance).
Local short-term power forecasts	Describes forecast for short-term future local power production and consumption. Data owned by the owner of energy production and consumption equipment (inverters and solar panels). Data may have privacy issues (GDPR compliance).
Compressed Local Power forecasts and schedules	Compressed versions of Local Power forecasts and schedules are sent to the cloud. Data owned by the owner of energy production equipment (inverters and solar panels). Data may have privacy issues (GDPR compliance).
Real time power (PV, Storage, grid meter, local consumption)	Created/harvested by local data collection block. Data owned by the owner of energy production equipment (grid meter, consumption, storage and solar panels). Data may have privacy issues (GDPR compliance). Sent to the cloud via Rest API interface.
Meteorological data forecasts	Retrieved from 3rd party data services on the cloud, and sent from the cloud to the edge for short-term PV and load prediction with the requested granularity. Created by weather forecaster, and harvested by local cloud communication block. This forecast is owned by 3rd party data service – does not have privacy issues

[1] To be finalized based on the current work in progress of Sirris on forecasting and optimizing methods on the edge device

Short-term battery schedules	As the output of the local optimization engine on the edge, short term charge/discharge commands are sent to the asset controller (Battery management system or inverter/logger)

2.3 Use case 2: Secure Internet provisioning (UC owner: NOS)

The NOS use case makes use of network traffic routers (CPE device) installed at each client's premises acting as Edge Node devices. Most of the MIRAI application blocks run inside this edge device. The remaining blocks all run in a cloud backend controlled by NOS.

The router/CPE are low power x86 embedded computers, with limited memory (xxx RAM and xxx Flash), running a version of Linux optimized for client router applications. The device supports development of C and C++ applications, as well as Java and python.

The cloud backend runs on virtual computers, that may be transferred at any time between cloud platform providers (Amazon AWS, Microsoft Azure, etc.). The application itself consists of data gathering, processing and transmitting blocks on the CPE, and further data processing blocks on the cloud.

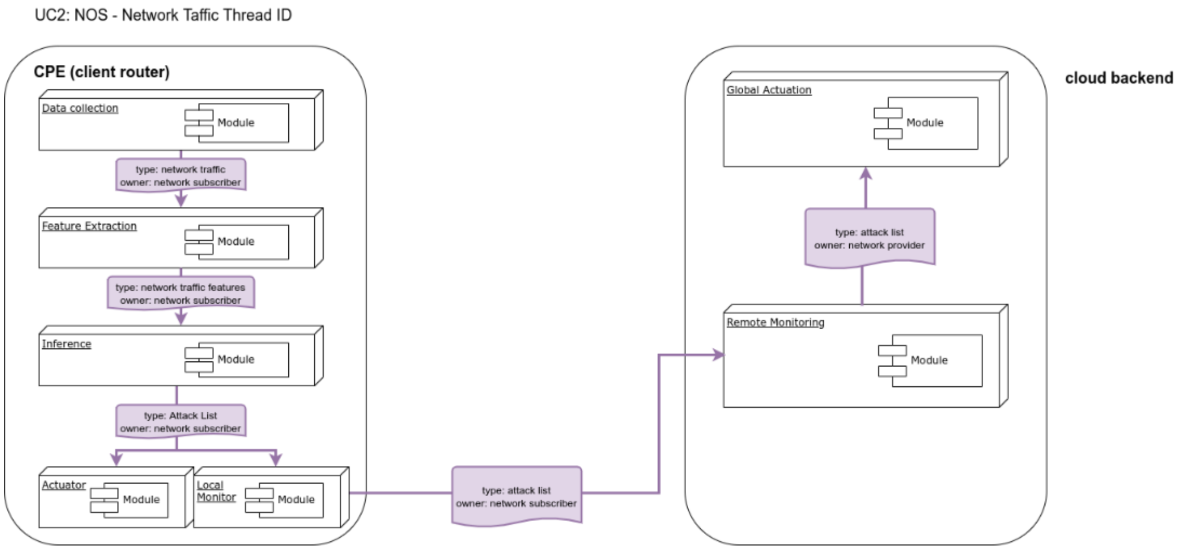


Figure 2-3-Component Diagram and Data Flows (UC 2)

Block	Description
Data Collection	Block collects signatures of traffic going through the router. Both outgoing and incoming packets are tracked. Only the first packets of each stream (TCP, UDP) are captured, since processing of further packets is offloaded to another onboard processor optimized for fast routing of packets and to which the main CPU does not have access to.
Feature extraction	Block extracts the main features of each packet (e.g. originating IP addresses and ports, packet type, protocol, …), and packet stream.
Inference	Block running local ML algorithms, used to identify possible malicious traffic.
Actuator	Block that will act locally on the information obtained from the inference engine - e.g. if malicious traffic is identified, then traffic stream is aborted.

Local Monitor	Block that compiles results of local inference at each edge device, and transmits results to centralized cloud.
Remote Monitoring	Block that compiles results from remote monitoring running on each client device, and makes inferences based on the aggregated results.
Global Actuation	Block that makes changes to the NOS routers at upper layer network level in order to minimize consequences of identified malicious traffic.

Dataflows	Description
Network Traffic (local)	Network traffic flowing locally on each CPE device (first packets of each network stream). This data contains information that is considered private by the NOS client using the CPE device, including information that may potentially identify that same client.
Network Traffic Features (local)	The features of the network traffic flowing locally at each CPE device (e.g. originating IP addresses and ports, packet type, protocol, …). This data contains information that is considered private by the NOS client using the CPE device, including information that may potentially identify that same client.
Attack List (local)	List of inferred attacks identified in the traffic passing the local router/CPE, sent from the local inference block to the local actuation and local monitoring blocks. This data contains information that may be considered private by the NOS client using the CPE device, even though it is unlikely to include information that may identify that same client.
Attack List (remote)	List of inferred attacks identified in the traffic passing the local router/CPE, sent from the local monitoring block to the remote monitoring block. This data flow consists of the same information as the previously identified data flow, which means that it also contains information that may be considered private by the NOS client using the CPE device, even though it is unlikely to include information that may identify that same client. Notice that this data flow travels over an open network.
Aggregated Attack List	List of inferred attacks currently passing the NOS network, obtained by aggregating (and potentially running inference algorithms) all the local attack lists. Notice that this data may travel over an open network in case the Global Actuation and the Remote Monitoring blocks are executed on distinct virtual machines of the cloud provider. Even though the data flow contains aggregated data, the data itself has not been anonymized so it may still potentially contain data that may be considered private by the many NOS clients, potentially including information that may identify individual clients.

2.4 Use case 3: Traffic management (UC owner: Macq)

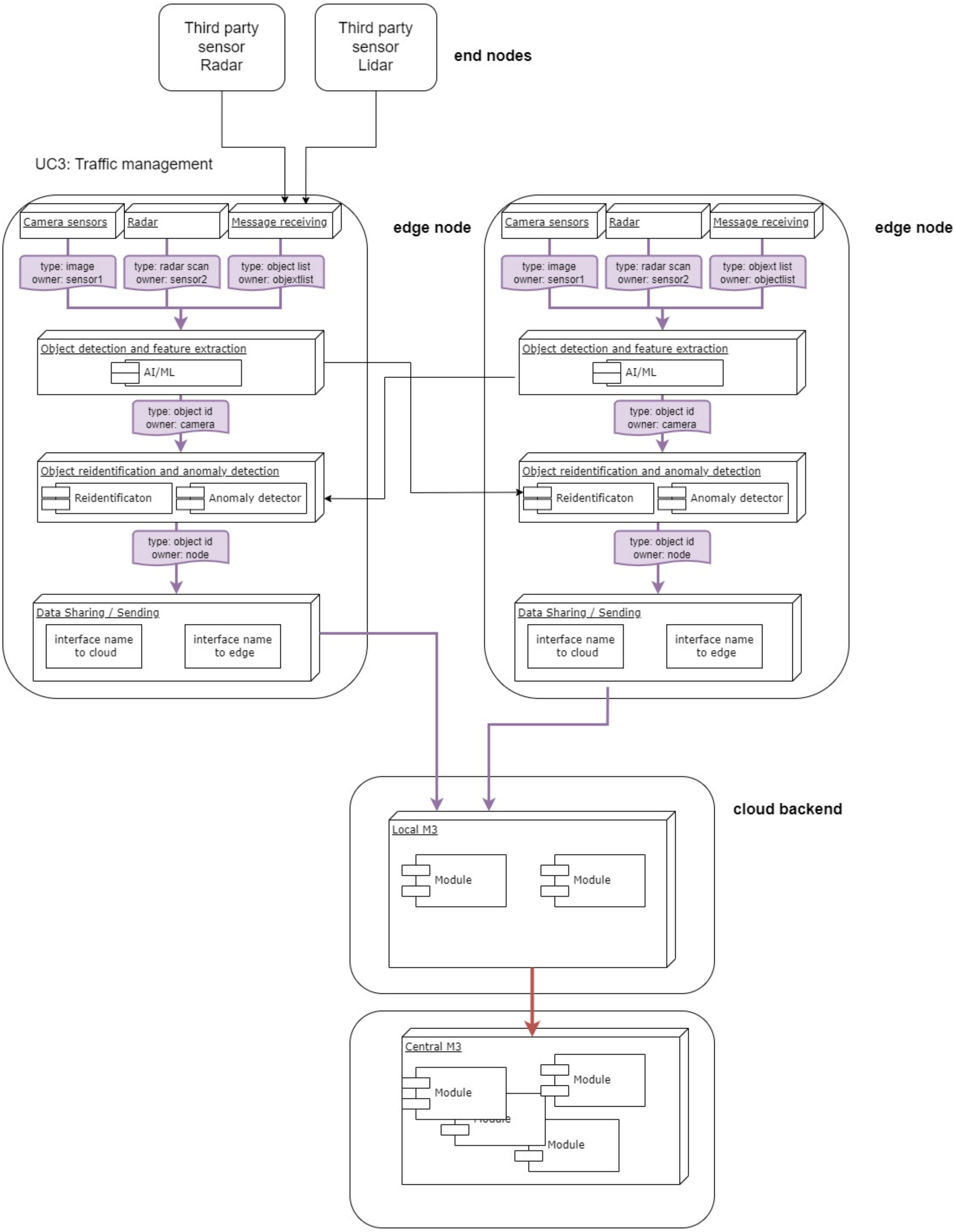


Figure 2-4-Component Diagram and Data Flows (UC 3)

Block	Description
Camera sensor	Depending on the camera type the Macq cameras of the QCAM series have 1 or 2 visual sensors. They can be Color, B&W or polarized.

Radar sensor	This sensor to be integrated in the camera is under development.
Third party sensors	Third party sensors like Radar, Lidar, Sound can be integrated via Ethernet or USB
Object detection and feature extraction	This is the main AI/ML block of the camera. We distinguish two purposes of this block: detection of objects (Vehicles - car, truck ..., and Vulnerable Road Users - pedestrians, cyclists …),and detection of features. Because features are most likely found in the same lower-level layers as this used for object detection, we keep it in one block. The ML algorithms are mostly implemented in the GPUs. Going forth and backwards between CPU and GPU memory is not efficient.
Reidentification	Based on the features and or object list determine that objects detected by two sensors are the same. The features are attributes of the objects. They are not necessarily human-interpretable. The objects list contains position, speed, size etc.
Anomaly detection	Based on position, speed, trajectory, posture, gate, interaction, … Note there is also anomaly detection of camera attributes (vibration, temperature, disk occupation, …) that can be used for predictive maintenance but that is a different execution chain;
Data sharing sending	Communication to M3 backend. It can also be communication to a data fusion AI box. Not shown on the diagram.

Dataflows	Description
Distribution of features	Horizontal communication between cameras
Object lists	Vertical communication between sensor boxes and cameras or AI boxes
Results	Communication from edge nodes (cameras) to backend servers (M3)

2.5 Use case 4: Water management (UC owner: Shayp)

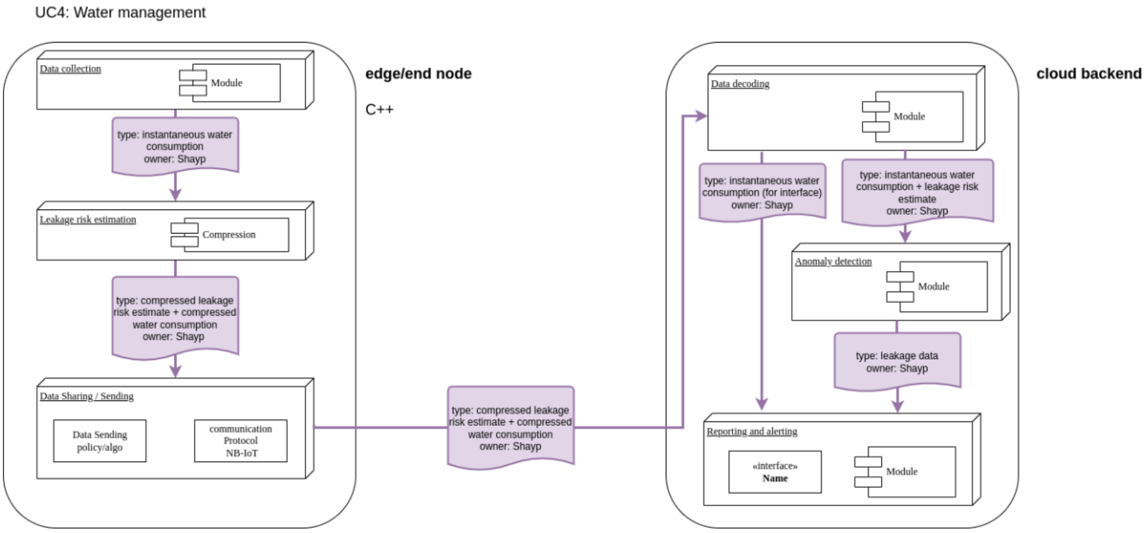


Figure 2-5-Component Diagram and Data Flows (UC 4)

Block	Description
Data collection	Collect water consumption data from the water meter through a pulse emitter. The pulses are counted in windows of 30 seconds and the resulting counts are stored into microcontroller memory.
Leakage risk estimation	Water consumption data are compressed following 2 different compression algorithms. The ratio of the lengths of the resulting compressed sequences gives the leakage risk estimate. This ratio along with the compressed water consumption data are then encoded for compression purpose to fit in the message payload.
Data Sharing / Sending	Send the compressed water consumption data and the compressed leakage risk estimate to the cloud through NB-IoT.
Data decoding	Decode/Decompress the water consumption data and the leakage risk estimate.
Anomaly detection	Apply anomaly detection cloud algorithm to output leakage data (amount and time distribution of the water that leaked).
Reporting and alerting	Report the water consumption data and the leakage data through a web application interface accessible to the customer. Also alert the customer in case of a consumption anomaly through SMS/email.

Dataflows	Description
Instantaneous water consumption	The water consumption sensed at the water meter as a time series aggregated in 30-second windows. This data is owned by Shayp and contains information private to Shayp's client.
Compressed leakage risk estimate	Leakage risk estimate is a ratio to be confronted to a given threshold to decide for an anomaly in the water consumption. It is compressed to fit into 3 bytes (1 byte for the integer part and 2 bytes for the fractional part). This data is owned by Shayp and contains information private to Shayp's client.

Compressed water consumption	Water consumption data compressed using Fibonacci compression algorithm and then further compressed by concatenating the bits of each 30-second window data into one sequence of bits instead of using 1 byte for each 30-second window data (typically, each of these data requires less than 8 bits). This data is owned by Shayp and contains information private to Shayp's client.
Instantaneous water consumption (for interface)	Water consumption as a time series aggregated in 1-minute windows. This data is owned by Shayp and contains information private to Shayp's client.
Leakage data	Leakage data as a time series aggregated in 1-minute windows. This data is owned by Shayp and contains information private to Shayp's client.

2.6 Use case 5: Continuous auto configuration of industrial controllers at edge (UC owner: Eliar & Enforma)

In data sharing between machines, it is not desired that the data be stored in one place and pulled by all machines. It has been decided to use the Publisher-Subscriber (Pub/Sub) structure. In Pub/Sub structure, subject or content-based filters are made so that the subscriber cannot reach every message, but it is thought that such a filtering is not needed in our topology yet.

There are solutions for Pub/Sub as SaaS (Software as a Service), as well as messaging middleware such as Apache Kafka, Pulsar, ActiveMQ and RabbitMQ. Kafka is ideal for big data use cases that require the best throughput, while RabbitMQ is ideal for low latency message delivery, guarantees on a per-message basis, and complex routing. Since it is thought that filters can be used in the future for simulation, RabbitMQ, which is open source among the middleware, has been preferred. The data we use for our use case cannot be classified as big data. Additionally, routing is essential to our use case, and Kafka cannot handle routing.

RabbitMQ is a messaging broker that supports many messaging protocols and can be used in distributed and unified configurations. Since it is widely used, almost every programming language has a client. In the Pub/Sub logic in RabbitMQ, the following steps are essential:

● "Producer" sends a message to Exchange,

● Message sent to Exchange is stored in queue for buffering,

● "Consumer" receives the message in queue and the message is deleted from queue.

Exchange is the message forwarding tool defined by the virtual host in RabbitMQ. It forwards the message to the relevant queue with the specified routing key or binding. Producer often doesn't know whether the message will reach the queue as it doesn't send its message directly to the queue and there is Exchange as a layer between it and the queue. If there is no queue for the message, Exchange tags the message as unrouteable and deletes it from the system. There are Exchanges of Direct, Topic, Headers, Fanout types in RabbitMQ. The process that each of them does is different and it is an approach that should be considered and decided according to necessity in forwarding the messages to the relevant queue.

- **Fanout Exchange:** Copies and routes a received message to all queues that are bound to it regardless of routing keys or pattern matching as with direct and topic exchanges. The keys provided will simply be ignored.

- **Direct Exchange:** Delivers messages to queues based on a message routing key. A message goes to the queue(s) with the binding key that exactly matches the routing key of the message.
- **Header Exchange:** The header sent in the message and the value bound to the queue are matched over the "x-match" key and forwarded to the matches.
- **Topic Exchange:** Route messages to one or many queues based on matching between a message routing key and the pattern that was used to bind a queue to an exchange.

In order for the message shared by one machine to be received by all the machines, it has been decided to use Fanout, one of the Exchange types. One downside of Fanout that should be considered is that it shares the message to all queues so that it can be delivered to all Consumers. Suppose there are 10 machines in the simulation, when 1 machine sends a message, it will send the message to the queue of 10 machines including itself. In this case, considering that all 10 machines send a message per second; n^2, i.e. 100 messages will be shared per second. Considering that there are 100 machines, the number of messages to be shared per second will be 10,000. Since Eliar has an average of 30 to 35 machines in factories, and they will communicate once per second, this would translate to 1,225 messages being sent per second. Although this number of messages isn't high, it should be examined in terms of bandwidth. This requires careful selection of the attributes to be sent. Besides being scalable, RabbitMQ is considered to be located on a separate server from the machines & Telescope.

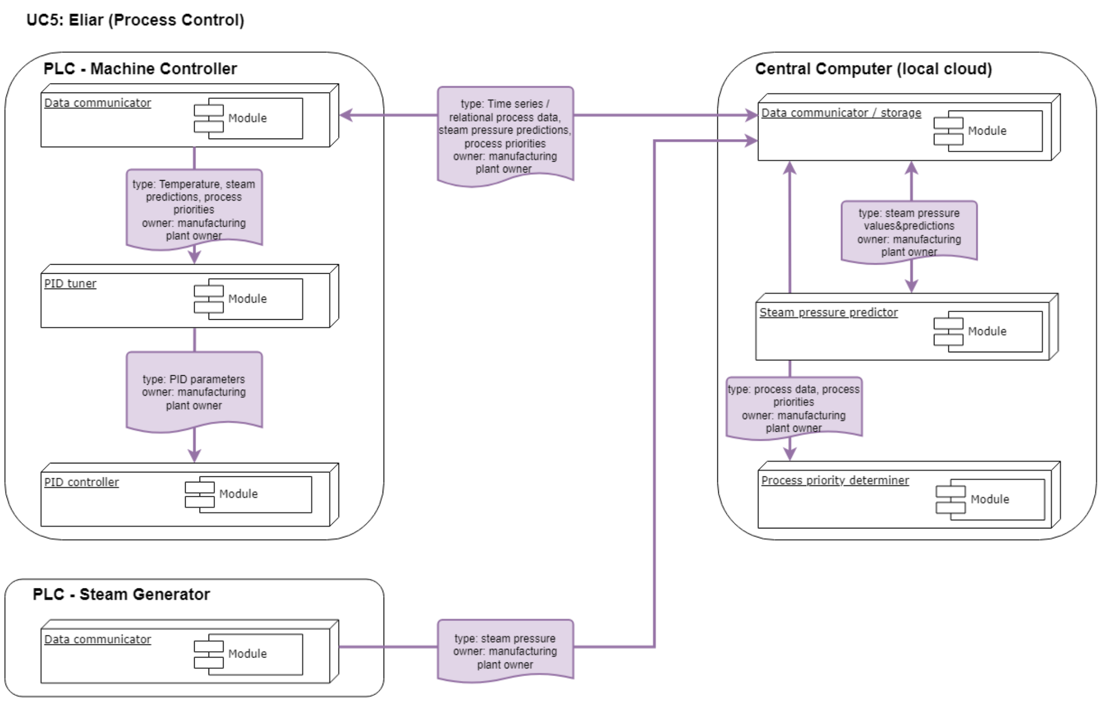


Figure 2-6-Component Diagram and Data Flows (UC 5)

Block	Description
Data Communicator / Storage	Collects and shares corresponding data like IOs, set points, process variables, counters.
PID Controller	Creates a control output based on PID control method by using given PID parameters like proportional gain, integral time, integral limits.
PID Tuner	Tunes PID parameters by using reference temperature values, steam predictions and process priorities.

Steam Pressure Predictor	Predicts steam pressure by using past steam pressure data for a predetermined horizon.
Process Priority Determiner	Determines process priorities based on process dynamics, recipes and other configurable rules.

Dataflows	Description
Steam Pressure	This data describes the pressure of the steam source that supplies super-heated steam to the dyeing machines to increase dye liquid temperature. Steam generator sends this data to local cloud. Data is owned by the manufacturing plant. Data does not have privacy issues.
Dyeing Process Data	This data describes the dyeing process. It includes analog and digital input & output values, set points, process variables, process events, consumptions. Machine controllers send/receive this data to/from local cloud. Data is owned by the manufacturing plant. Data does not have privacy issues.
Steam Pressure Predictions	This data describes the prediction of the steam source pressure. It is calculated by steam pressure predictor and stored in the local cloud to be sent to machine controllers. Data is owned by the manufacturing plant. Data does not have privacy issues.
Process Priorities	This data describes the process priorities. It is calculated by process priority determiner and stored in the local cloud to be sent to machine controllers. Data is owned by the manufacturing plant. Data does not have privacy issues.
PID Tuner Inputs	Dye liquid temperature values, steam pressure predictions and process priorities are sent to the PID tuner from machine controller. Data is owned by the manufacturing plant. Data does not have privacy issues.
PID Parameters	This data describes the PID controller parameters that are used to calculate control inputs. The PID tuner sends them to the PID controller. Data is owned by the manufacturing plant. Data does not have privacy issues.

3 Data Sharing Considerations per Use-case

3.1 Use case 1: Distributed renewable energy systems (UC owner: 3E)

Data Flow	Restrictions
Instantaneous power production	May not be made public without first being anonymized
Weather forecast	If the weather forecast is obtained as a commercial (paid for) service, data may not be shared with third parties
Local Power production forecast	May not be made public without first being anonymized
Compressed Local Power production forecast	May not be made public without first being anonymized

3.2 Use case 2: Secure Internet provisioning (UC owner: NOS)

Data Flow	Restrictions
Network Traffic (local)	Besides NOS, only the client on which the traffic originated may have access to the information contained in this data flow. NOS is allowed to make limited use of this information due to the provisions on the service contract signed between NOS and the client.
Network Traffic Features (local)	Besides NOS, only the client on which the traffic originated may have access to the information contained in this data flow. NOS is allowed to make limited use of this information due to the provisions on the service contract signed between NOS and the client.
Attack List (local)	Besides NOS, only the client on which the traffic originated may have access to the information contained in this data flow. NOS is allowed to make limited use of this information due to the provisions on the service contract signed between NOS and the client.
Attack List (remote)	Besides NOS, only the client on which the traffic originated may have access to the information contained in this data flow. NOS is allowed to make limited use of this information due to the provisions on the service contract signed between NOS and the client.
Aggregated Attack List	Only NOS may have access to the information contained in this data flow, as it potentially contains information from any and all NOS clients.

3.3 Use case 3: Traffic management (UC owner: Macq)

Data Flow	Restrictions
Distribution of features	Only allowed between statically configured cameras in local network. Secured MQTT protocol (To be confirmed)
Object lists	Only allowed with statically Secured MQTT protocol.
Results	All communication is TLS protected. Only communication with dedicated backend servers in a local network or VPN is allowed. Very privacy sensitive: ANPR data Image

3.4 Use case 4: Water management (UC owner: Shayp)

Data Flow	Restrictions
Instantaneous water consumption	Besides Shayp, only the client owner of the water meter from which this dataflow originates may have access to the information contained in this data flow. However, it is not relevant for the client to access this data at this stage of the process as it will be nicely presented through an interface further down the process.
Compressed leakage risk estimate	Only Shayp may have access to the information contained in this data flow as no information on the way data is compressed/encoded for later sending is to be publicly available.
Compressed water consumption	Only Shayp may have access to the information contained in this data flow as no information on the way data is compressed/encoded for later sending is to be publicly available.
Instantaneous water consumption (for interface)	Besides Shayp, only the client owner of the water meter from which this dataflow originates may have access to the information contained in this data flow.
Leakage data	Besides Shayp, only the client owner of the water meter from which this dataflow originates may have access to the information contained in this data flow.

3.5 Use case 5: Continuous auto configuration of industrial controllers at edge (UC owner: Eliar & Enforma)

Data Flow	Restrictions
Steam Pressure	Besides Eliar & Enforma, only the end-user (manufacturing plant) on which the textile processes are held may have access to the information contained in this data flow. Eliar is allowed to make limited use of this information due to the provisions on the service contract signed between Eliar & Enforma and the end-user.
Dyeing Process Data	Idem

Steam Pressure Predictions	Idem
Process Priorities	idem
PID Tuner Inputs	idem
PID Parameters	idem

4 Generic Data Sharing Policies Description Framework

For the MIRAI project we propose a generic data sharing policy framework that allows a clear definition of the restrictions placed on the data sharing for each use case. Note that this data sharing policy framework does not include any entity to restrict access to or sharing of data. Such entity would imply a specific architectural constraint. For the sake of flexibility, the MIRAI reference architecture left this responsibility to each use case. The purpose of this data sharing policy is just to lead the system designer to explicitly state the data sharing properties.

The framework is based on the following concepts:

Data type: a type of data, such as instantaneous electrical power being produced.

Entity: An entity that may own data, or an MFBB instance being executed. For example, the owner of the power inverters and solar panels producing electricity, or the entity collecting and harmonising data from several electrical production installations.

Data flow type: A tuple of (data type, entity), defining the type of data and the owner of the data being transmitted between two MFBBs.

Data flow: an instance of data flowing between two MFBBs of a specific Data flow type.

Data processing block: An MFBB with a well-defined data flow type for input data, and data flow type for output data, executing under the control of a specific entity. The output type may have the same owner/entity as the input data flow type. The data processing block may also change the data owner/entity, for example in case of a data anonymizer block.

Data sharing policies: A map, defining which entities have (read) access permission of each specific data flow type, i.e. for each (data type, entity) tuple.

5 Data Sharing Policies per Use Cases

5.1 Use case 1: Distributed renewable energy systems (UC owner: 3E)

List of Data Types

Data Type	Description
Power/Energy flows	Power/Energy of all the local assets from PV, storage, grid meter, and consumption
Forecasting data	Including PV, consumption, and meteorological forecasts
Scheduled data	Results of the storage charge/discharge optimization engine either on the edge or on the cloud

List of Entities

Entity	Description
Prosumer	A C&I tertiary building with local assets and systems to produce, store, and consume electrical energy
3E	Software as a Service provider of Building Energy Management Systems

List of Data Flow Types

Data Flow Type	(Data type, Entity)	Description
FTP or Rest API	Power/Energy flows, prosumer	Csv file or time-stamped measurements
FTP or Rest API	Forecasting data, 3E	Csv file or time-series
FTP or Rest API	Scheduled data, 3E	Csv file or time-series

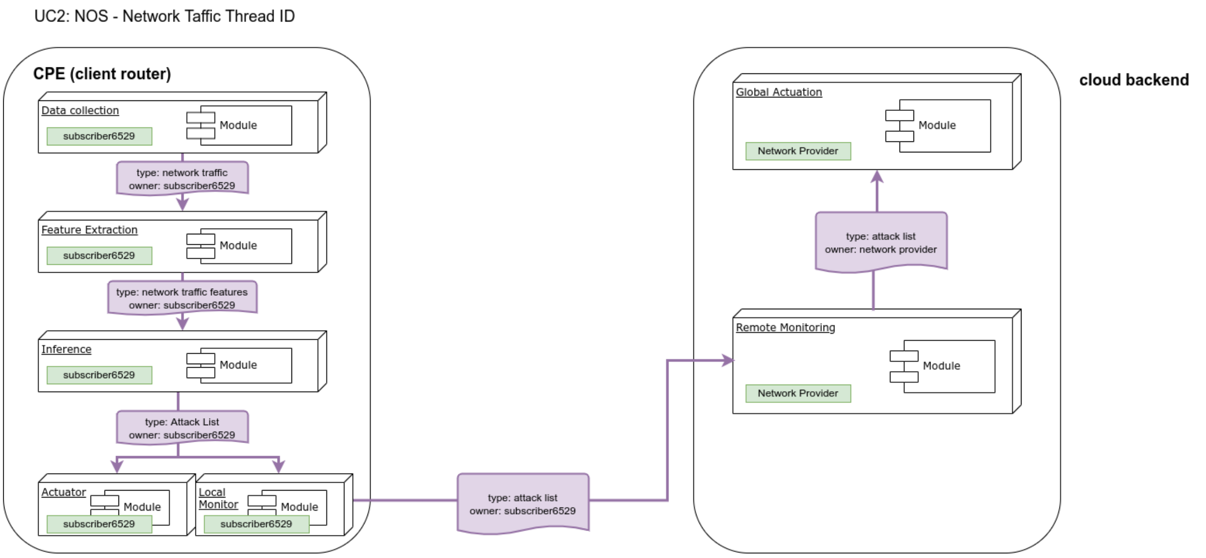
Data Sharing Policy Map

Data Flow Type	Entities (with read permission)
Power/Energy flows	3E, prosumer
Forecasting data	3E, prosumer
Scheduled data	3E, prosumer

5.2 Use case 2: Secure Internet provisioning (UC owner: NOS)

Mapping the NOS use case onto the MIRAI data sharing policies framework requires that each NOS client be defined as a (data owner) entity. NOS itself is also defined as an entity capable of producing and/or processing data.

All data processing MIRAI framework blocks running in the client's router (the CPE device) are considered to be executing under the permissions of the client. On the other hand, all data processing MIRAI framework blocks running in the cloud are considered to be executing under the permissions of NOS (I.e. the network provider).



List of Data Types

Data Type	Description
Network traffic	Network traffic (I.e. the first packets of each data stream)
Network Traffic Features	Features of network traffic (e.g. IP address, ports, protocols, ...)
Attack List	List of potentially malicious attacks/traffic

List of Entities

Entity	Description
Client XXX	Each NOS client (network subscriber) is considered a distinct entity in the MIRAI data sharing policy framework
NOS	The network provider

List of Data Flow Types

Data Flow Type	(Data type, Entity)	Description
Local Network traffic	(Network traffic, Client XXX)	
Local Network Traffic Features	(Network traffic features, Client XXX)	

Attack List	(Attack List, Client XXX)	
Aggregated Attack List	(Attack List, NOS)	

Data Sharing Policy Map

Data Flow Type	Entities (with read permission)
Local Network traffic	Client XXX, NOS
Local Network Traffic Features	Client XXX, NOS
Attack List	Client XXX, NOS
Aggregated Attack List	NOS

5.3 Use case 3: Traffic management (UC owner: Macq)

In the NextPerception project an Object List protocol on top of MQTT was defined.

List of Data Types

Data Type	Description
Image	Image as captured from the sensor
Feature	Patch of a ML layer allowing reidentification
Coordinate	Latitude, Longitude, Altitude. WGS84 coordinates on the ground.
Bounding Box	Rectangle around the detected object in the image
Velocity	Speed in m/s, Acceleration and Direction
RoadUserType	Enumeration type: Car, Truck, Bicycle, Pedestrian, …
RoadUserObject	ID, Timestamp,LastSeen, Type: array of RoadUserType, Position, BoudingBox, Velocity, CovarianceMatrix, ExistanceProbability, ObjectOrientatation,
DetectedObjectList	Number of objects and array of detected road user objects for one message. This may be all objects but not have to be. (e.g. due to reduced sample rate this might not include all objects)
SensorProperties	ID, Position, DetectionArea, FreeSpaceAddendum, UpdateRate, SensorType, MaxAmountOfObjects
ProtocolRootMessage	ProtocolVersion, DetectedObjectList, SensorProperties, SensorID

List of Entities

Entity	Description
Camera	Edge device with sensors
M3	Backend server on VPN or in the Cloud.
AI box	Edge device without sensors

List of Data Flow Types

Data Flow Type	(Data type, Entity)	Description
Result image	(Image, Camera)	Used as proof of event
Feature lists	(Feature, Camera)	Used for data fusion
Object lists	(ProtocolRootMessage, Camera)	Used for object fusion or final result
Video stream	(Sequence of images, Camera)	Used for camera setup

Data Sharing Policy Map

Data Flow Type	Entities (with read permission)
Video	Data Processor, Maintenace during installation
Result image	Data Processor using M3
Feature lists	Camera and AI box
Object List	Camera, AI box and M3

5.4 Use case 4: Water management (UC owner: Shayp)

List of Data Types

Data Type	Description
Water consumption	Instantaneous water consumption is detected from the flow inside the water meter.
Leakage risk estimate	Leakage risk estimate computed at the edge as a ratio to be compared to the leak threshold.
Leakage data	Leakage data computed at the cloud based on the water consumption and the leakage risk estimate.

List of Entities

Entity	Description
Shayp	The service provider
Water consumer client	

List of Data Flow Types

Data Flow Type	(Data type, Entity)	Description
Instantaneous water consumption	(Water consumption, Shayp)	
Compressed leakage risk estimate	(Leakage risk estimate, Shayp)	
Compressed water consumption	(Water consumption, Shayp)	
Instantaneous water consumption (for interface)	(Water consumption, Shayp)	
Leakage data	(Leakage data, Shayp)	

Data Sharing Policy Map

Data Flow Type	Entities (with read permission)
Instantaneous water consumption	Shayp
Compressed leakage risk estimate	Shayp
Compressed water consumption	Shayp
Instantaneous water consumption (for interface)	Water consumer client, Shayp
Leakage data	Water consumer client, Shayp

5.5 Use case 5: Continuous auto configuration of industrial controllers at edge (UC owner: Eliar & Enforma)

List of Data Types

Data Type	Description
Steam Pressure	Pressure data of steam source of the manufacturing plant
Dyeing Process Data	Data of dyeing processes like IOs, set points, process values etc.
Steam Pressure Predictions	Predictions of steam pressure
Process Priorities	Process priorities like low or high of dyeing processes
PID Tuner Inputs	Input data of the block that tunes the PID parameters
PID Parameters	Output data of PID tuner block which constitutes the PID parameters

List of Entities

Entity	Description
Eliar	The Service & Technology Provider
Enforma	Technology Provider
Manufacturing Plant	The End-User

List of Data Flow Types

Data Flow Type	(Data type, Entity)	Description
Steam Pressure	(Steam pressure, Manufacturing plant)	
Dyeing Process Data	(Dyeing process data, Manufacturing plant)	
Steam Pressure Predictions	(Steam pressure predictions, Manufacturing plant)	
Process Priorities	(Process priorities, Manufacturing plant)	
PID Tuner Inputs	(PID tuner inputs, Manufacturing plant)	
PID Parameters	(PID parameters, Manufacturing plant)	

Data Sharing Policy Map

Data Flow Type	Entities (with read permission)
Steam Pressure	Manufacturing plant, Eliar, Enforma
Dyeing Process Data	Manufacturing plant, Eliar
Steam Pressure Predictions	Manufacturing plant, Eliar, Enforma
Process Priorities	Manufacturing plant, Eliar
PID Tuner Inputs	Manufacturing plant, Eliar
PID Parameters	Manufacturing plant, Eliar

6 Final Remarks

This document lists the data sharing policies and processes identified for the MIRAI Framework architecture, produced under task T3.2 of project ITEA3 MIRAI. The output of this task will feed subsequent tasks of WP3, whose purpose is to implement the MIRAI framework. This task took place in close relationship with task T3.1, meant to design the “Secure Reference Architecture”.

Applying the data sharing policies framework to each use case shows that a separate write access permission map is not required.