Deliverable

D7.3 Verification results of the blocks and reference nodes

Deliverable Id:	D7.3(1)
Deliverable Name:	Verification results of the blocks and reference nodes – first version, to be updated in M36
Status	Submitted
Dissemination Level:	Public
Due date of deliverable:	31/12/2022
Actual submission date:	31/12/2022
Work Package:	WP7
Organization name of lead contractor for this deliverable:	AVL
Author(s):	Bernhard Peischl, AVL
Partner(s) contributing:	Alfonso Gonzalez Gil, ZYLK Bekim Chilku, SIEM Inaki Paz, LKS Michael Gautschi, ACP Martin Rivas Caneiro, PROI Tomislav Bukic, AVL
Reviewers:	Ana Patricia Bautista, IKERLAN Susanna Pirttikangas, OULU Lauri Loven, OULU

Abstract: In this deliverable we demonstrate on how to refine systemic properties referred to as FRACTAL features using the FRCATAL building blocks. Each FRACTAL feature is refined in terms of a three-layered architecture addressing the application-, the service-orchestration- and node level. The deliverable reports on recently obtained KPIs capturing the technical- and the business advantages of the FRACTAL features.

This project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 877056

Co-funded by the Horizon 2020 Programme of the European Union under grant agreement No 877056.

Contents

1 History

Version	Date	Modification reason	Modified by
V0.1	17/11/2022	Document structure proposal	Bernhard Peischl
V0.2	01/12/2022	Preliminary content for UC2	Bernhard Peischl, Tomislav Bukic
V0.3	13/12/2022	First version of the introduction	Bernhard Peischl
V0.4	17/12/2022	Extended description of UC2	Tomislav Bukic
V0.5	17/12/2022	Description of UC4	Bekim Chilku
V0.6	17/12/2022	Description of UC3	Michael Gautschi
V0.7	17/12/2022	Harmonization and summary	Bernhard Peischl
V0.8	18/12/2022	Summary and Abstract	Bernhard Peischl
V0.9	19/12/2022	Description UC1	Martin Rivas, Alfonso Gonzales
V1.0	19/12/2022	Final consolidation and formatting	Bernhard Peischl
V2.0	30/12/2022	Addressing suggestions from internal reviews	Bernhard Peischl

2 Summary

This deliverable demonstrates how to refine top level systemic properties referred to as FRACTAL features using the dedicated building blocks provided by the technical work packages. It builds on Deliverables D7.1 (essentially the plan for integration and verification including technical and business KPIs) and D7.2 (in which we elaborate the implementation of four lower TRL use cases). D7.3 implements an application pull, i.e., it reflects the needs of industrial use cases and as such refines five dedicated FRACTAL features: (1) cognitive awareness, (2) cloud connectivity/communication, (3) adaptability, (4) security, (5) low power. Further use cases 2 and 3 contribute to (6) openness.

Relying on the concrete needs of the use cases 1 to 4, throughout the implementation phase we suggested a top-level categorization for the FRACTAL components (see D7.2, Section 3.1 FRACTAL High Level Architecture). D7.3 demonstrates how the different components are used to refine the systemic FRACTAL features using the layered architecture style. According to this top-level concept, each use case groups the employed FRACTAL components into three layers (application-, orchestration- and node-layer). In this way we capture the view of the overall system and its systemic properties while providing enough detail to understand the roles and responsibilities of the individual layers and the relationship between them.

Each use case illustrates the specific instantiation of (a number) of pivotal FRACTAL feature(s) and afterwards elaborates the experimental setup on which the assessment is carried out. The quantitative assessment is carried out from the technical and a business point of view in terms of tangible KPIs. Finally, each use case discusses its specific path to exploitation.

This deliverable is the minimal viable version of D7.3 to be submitted in M36.

3 Introduction

In this deliverable we demonstrate how to refine the systemic properties (FRACTAL features) with the help of dedicated components (building blocks) developed in the technical work packages (WP3 to WP6). Deliverable D7.3 builds on D7.1 (essentially the plan for integration and verification of the components) and D7.2 (the implementation of the use cases) and focuses on the assessment of the technical and business KPIs for the use cases 1 to 4. D7.1, D7.2 and D7.3 together implement an application pull (i.e., the industrial needs act as a driver for the R&D activities) and as such the use cases 1 to 4 follow the same pattern:

Each use case addresses specific FRACTAL features. As these systemic properties are abstract concepts, first, each use case illustrates the specific feature to be demonstrated. D7.3 covers six FRACTAL features: (1) cognitive awareness, (2) cloud communication, (3) adaptability, (4) security, (5) low power and (6) openness (see deliverable D7.1 for a mapping between use cases and FRACTAL features). The notion of fractality is a composition of the various features including the communication between the FRACTAL cloud node(s) and the FRACTAL edge nodes. D7.1 outlines the meaning of the various features and aligns them with the use cases 1 to 4, D7.2 elaborates on the specific interpretation of the main feature in each use case.

The building blocks that have been used to refine the specific feature are listed alongside with the top-level layer for every use cases. The idea behind this is that the FRACTAL components with similar functionalities are organized in horizontal layers. The lowest layer deals with providing the computational nodes (node layer), the layer above provides abstractions to orchestrate the various services (service-orchestration) and the top-layer (application) hosts application specific components. In this way the layered architecture style abstracts the view of the system while providing enough detail to understand the roles and responsibilities of the individual layers and the relationship between them.

As the deliverable comes up with technical and business KPIs, each use case elaborates the specific setup of the demonstrator on which the assessment has been carried out such as data sets being used, specific assumptions under which the experiments have been carried out, third party systems being used or known limitations of the overall setup.

The quantitative assessment of the advantages is carried out from a technical and business point of view. Thereby each FRACTAL feature is characterized in terms of concrete quantitative technical and business KPIs.

D7.3 further illustrates a path to exploitation for every use case briefly sketching the value proposition alongside the addressed stakeholder groups for the future business services being offered.

This deliverable is a minimal viable version of D7.3. According to the extension of task 7.2 and 7.3 a final version of the deliverable with (where appropriate) updated KPIs will be provided in month 36.

4 UC1 – Engineering and Maintenance Works

4.1 Demo I - UAV supervision of critical structures

4.1.1 Addressed FRACTAL features

4.1.1.1 FRACTAL feature Cognitive Awareness

The UAV will be controlled by an experienced operator, and the designed AI system will automatically process and analyze the images to gain knowledge of structural defects that may pose a future hazard to the construction. The designed system is based on a Convolutional Neural Network (CNN) with U-Net architecture and residual connections. In addition, the system has been improved by using image augmentation techniques (translations, rotations, superposition of different textures, etc.) and a manually labeled dataset. The training of the model has been done offline and several variations of the proposed architecture have been tested to better adapt the system to the use case. Also, the inference will run online and in real-time on the edge. In addition, the images taken by the UAV and the inference made by our model will be uploaded to the cloud[1], so they can be used by a structural expert.

The system will be considered successful if it helps the technicians to better detect cracks. This can be measured in several ways. First, the quality of the Deep Learning Model will be evaluated with metrics such as IoU score and Categorical Cross Entropy. This model should reduce the costs and accidents inherent to "traditional" inspection methods that make use of special machinery. Moreover, it must help the technicians to better detect the number of cracks. Thus, it must result in a cognitive node that is aware of the defects of the structures (cracks) and introduces an improvement over a "traditional" method. This will be verified by technical metrics and, at a later stage, by the technicians who use this tool.

4.1.2 Building block integration and verification

There are different options to approach the correct algorithm to extract information from the images. Classical methods include approaches such as histogram thresholding (Otsu thresholding), analysis of local parameters and edge detection, growing algorithms and clustering methods[2]. However, in the recent years it has been shown the superior performance of deep convolutional neural networks (CNN) approaches. Concretely, three main deep learning approaches have been considered to face the instance segmentation problem[3]

[1] UC1 employs a commercial cloud platform.
[2] Zhu, Q. L. (2011). Image segmentation and major approaches. 2011 IEEE International Conference on Computer Science and Automation Engineering, (S. 465-468).
[3] Terzopoulos, S. M. (2020). Image Segmentation Using Deep Learning: A Survey. Von Arxiv.org: https://arxiv.org/abs/2001.05566

- U-Net based architectures: This type of CNNs includes encoder-decoder structures that allow extracting more complex features of the image, and residual connections that result in a better propagation of the information through the neural network.
- Multi-scale models (FPN): The main advantage of this method is the great performance at extracting features at different scales.
- Attention-based models (R-CNN): These models propose a region of the image in which the object may be located, and a posterior segmentation process is carried out.

The selected algorithm is a deep learning model based on a convolutional neural network (CNN). In order to solve the segmentation problem, the architecture of the deep neural network is U-Net (and ResNet), with a custom loss-function (combination of Categorical Cross entropy and IOU). The deep learning model is initially trained with a dataset generated and labelled in an early stage. In addition, the model has the capability to improve its performance, by being retrained with new images collected during the lifecycle of the system.

One of the biggest challenges of the UC1 was the insight generation on the edge (on the UAVs). Deep neural networks are expensive algorithms in terms of consumption of resources such as energy and computing power. Thus, in order to achieve a good performance, special hardware must be used. Three options have been considered:

- Xilinx-VERSAL: Xilinx processor with ACAP (Adaptable Compute Acceleration Platform) architecture (on the same chip there are ARM processors, an FPGA and a vector calculation acceleration unit (inference with AI models)).
- PULP: PULP is an open hardware project developed by the universities of Bologna and ETH Zürich. The PULP project provides the microprocessor architecture (RI5CY), and external companies develop chips with this architecture.
- Development of the prototype with other microcontrollers: When making a working proof of concept, other microcontroller options may be better suited for this UC in terms of price and ease of development. However, notice that this may not be the best option for production.

The commercial node Xilinx VERSAL ACAP was selected as hardware for the processing of the images. In addition, due to problems with the integration of the prototype board because of its size, there will be no direct communication between the drone and the processing environment, but the images will be downloaded for subsequent inference on the VERSAL board. Initially, the inference tests will be performed on a Jetson Nano board, since the VERSAL board is not available, and then the same solution will be developed on the VERSAL board to observe the performance improvement between the two boards.

The way to achieve the process of the model training, inference and re-training, is by using the following components from the FRACTAL platform:

feature	building block / component	layer
COGNITIVE AWARENESS	WP3T32-10 / VERSAL accelerator building blocks	application
	WP3T34-03 / VERSAL model deployment layer	application
	WP4T42-02 / VERSAL RPU access to AI acceleration	application
	WP6T61-01-03 / Data Analysis	application
	WP6T61-01-04 / Data pre-processing	application
	WP6T61-05 / ML/AI Tools	application

Table 1: FRACTAL building block in the various abstraction layers employed in UC 1.

4.1.3 Demonstrator setup

The goal of this demonstrator is the detection of cracks on reinforced concrete infrastructures. Concretely, this crack detection system will be used on structures such as bridges and constructions of limited access, on which it is difficult and expensive to carry out a human inspection. In order to get the insights of the cracks on concrete surfaces the following datasets have been considered:

- Deep Crack dataset – cracks on concrete infrastructures: https://github.com/yhlleo/DeepCrack
- Cracks on masonry: https://github.com/dimitrisdais/crack_detection_CNN_masonry
- To be studied: https://apps.peer.berkeley.edu/phi-net/

These datasets include images and their corresponding masks as shown below:

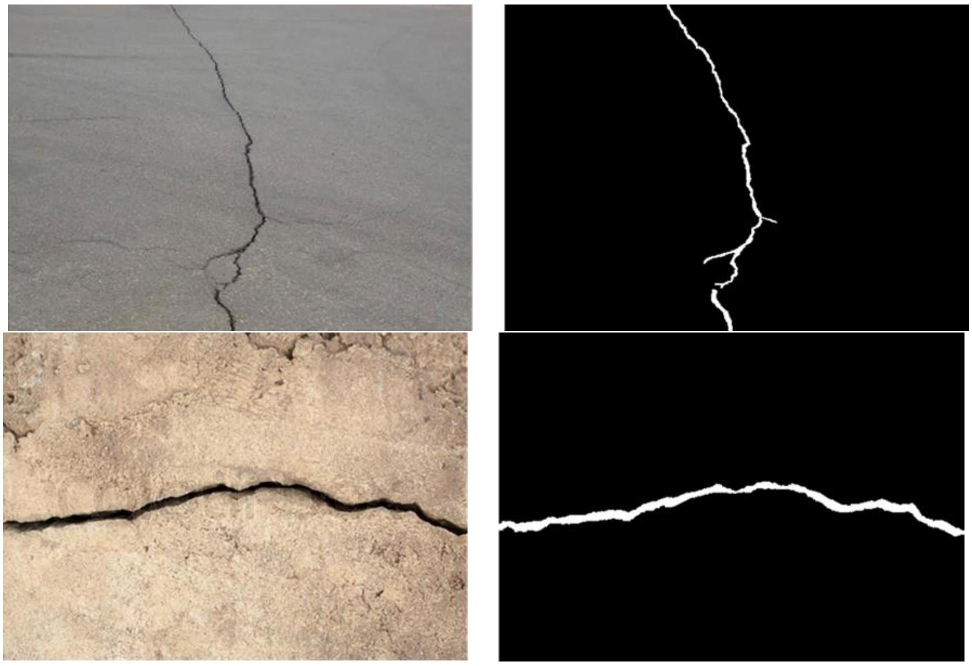

Figure 1: Original images and their created masks.

In addition to the datasets obtained on the Internet, images have been collected in different locations, so that the capture could be adapted to the training needs of the model used. In this sense, images have been obtained from the following locations:

Figure 2 Images collected with drone in different locations (in order from left to right and up to down): AUVISA, Ciudad Real (Spain), La Paz (Bolivia), A-31 highway (Spain), Níjar HST (Spain)

Once the images were collected, the model was trained. Because of its proven success on similar previous experiences, it has been decided to design a CNN with U-Net architecture. Furthermore, in order to make the CNN as configurable and adaptative as possible, a modular neural network has been designed, composed of the next sub-modules:

- **Residual block**: the residual block is the main and basic building block of our model. It is composed of convolutional layers followed by batch normalization and activation function layers. In order to allow a better propagation of different levels of information of the image, this block incorporates a skip connection (residual connection that connects the input and the output of the block).

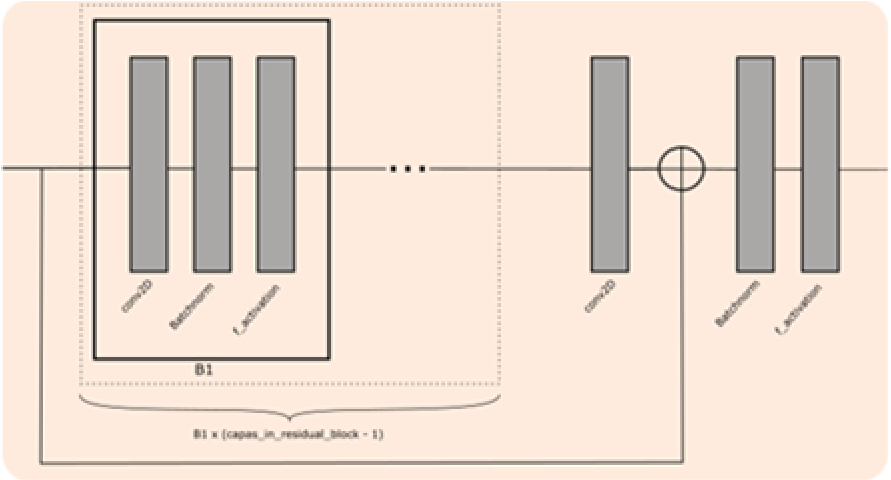

Figure 3: Residual block.

- **Encoder/Decoder block**: next, each level of the U-Net is composed by encoder/decoder blocks that gather several residual blocks into a single building unit. These blocks are connected between them via upsampling/downsampling layers and residual connections.

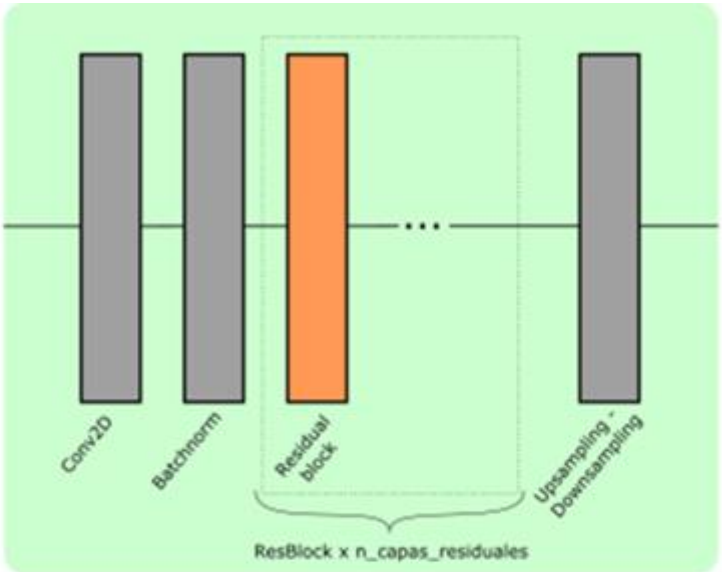

Figure 4: ED block.

- **Ensemble block:** the output of the CNN consists of an ensemble block. The main idea of this part of the model is that, starting from the information extracted from the neural network (final feature map), the combination of several small models will perform better than a single big model. Thus, several small models based on the combination of residual blocks are combined in parallel at the output of the model.

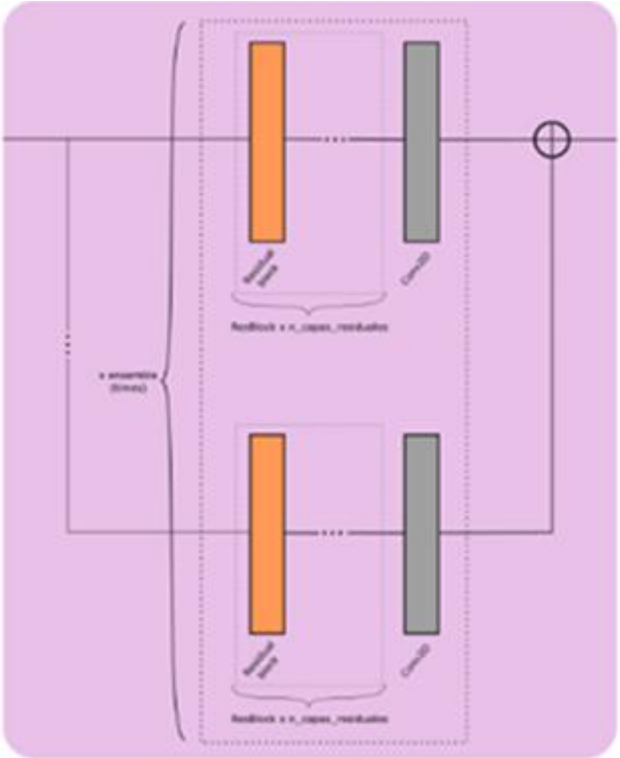

Figure 5: Ensemble block.

- **Model:** the final designed model will be based on a combination of the previous blocks. First, there is the U-Net architecture built with encoder/decoder blocks, followed by the ensemble block. The main advantage of this configuration is that the architecture of the neural network can be easily configured and optimized as if it was any other hyperparameter of the neural network.

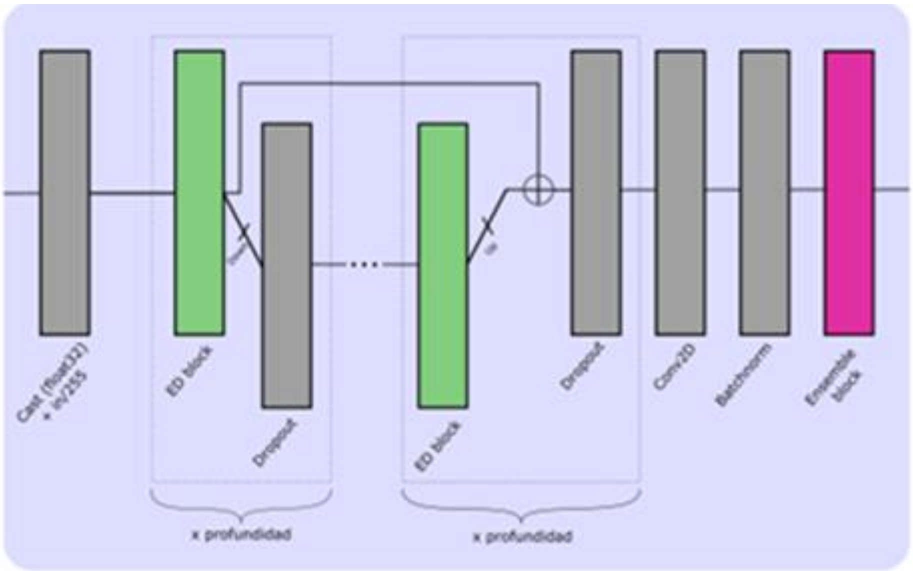

Figure 6: Neural Network (NN).

In order to obtain the best architecture of the neural network and the best configuration of parameters, a "pseudo" grid search has been carried out ("pseudo" because not all the possible combinations have been tested). In this way, several combinations of the components of the neural network and hyperparameters (learning rate, activation function) have been tested out, and the best one was selected. The following range of parameters has been considered:

- Input of RGB images of size 416x416x3. In some cases, in addition to the RGB images, the insertion of the images in HSV format (input of size 416x416x6) has been tested.
- Number of filters of the convolutional layers: The number of filters of the first layer ranges from 10 to 20.
- Depth: the tested depths of the U-Net part of the neural network range from 2 to 4 (2 to 4 downsampling/upsampling layers).
- Regularization: L2.
- The influence of batch normalization layers after each convolutional layer has been also tested.
- The number of layers of the residual block ranges from 1 to 3.
- The influence of downsampling by maxpooling vs downsampling by a convolutional layer with stride 2x2 has been tested.
- Filters of the convolutional layers of size 3x3 and 5x5 have been tested.
- Data augmentation: brightness augmentation and affine transformations.
- Loss function = cross-entropy + $\alpha \cdot$ IoU; with α ranging from 0 to 1.

The model has been trained with 2879 images, validated and tested on a set of 159 images. The dataset has been made with cropped images from the Deep Crack dataset without resizing. In addition, in order to get the best configuration of hyperparameters, all the models have been trained 20 epochs.

The first results were obtained after the hyperparameter optimization process, with the following parameters:

- Three downsampling/upsampling phases (depth of three).
- Initial input layer with 16 filters of size 3x3. Notice that after each downsampling layer, the number of filters is multiplied by 2, and after each upsampling layer, the number of filters is divided by 2.
- Batch normalization layers after each convolutional layer, no L2 regularization and ReLU activation function.
- The model is trained with data augmentation (brightness augmentation and affine transformations).
- Loss function = cross-entropy + 0.3 • IoU.
- The HSV image has been combined with the RGB image at the input (input of size 416 x416 x 6).

A model with these parameters has been trained 60 epochs on a dataset composed by 2879 images, obtaining an IoU score on the validation set of 0.7308. In Figure 9 it is shown the inference results of this model:

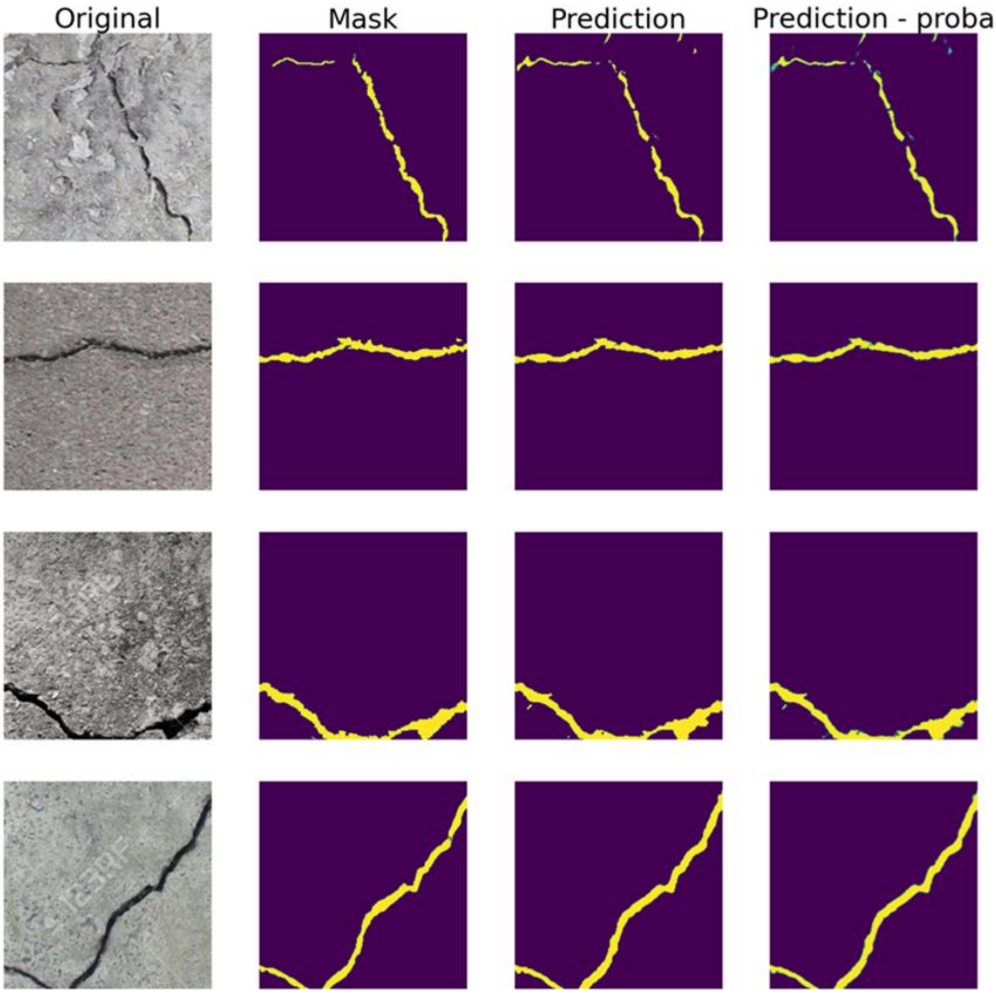

Figure 7: Result of the model trained 60 epochs.

The above figure shows the inference of our model on some images of the test set of the Deep Crack dataset. The first column shows the original image, the second shows the mask, the third one shows the prediction and the fourth column shows the predicted probability of a pixel of being a crack (this ranges from 0 to 1). As shown, our model success on the faced task. However, dark parts of the image are predicted as cracks even when they are not cracks (first image). This happens because in the original dataset, most dark parts of the image are cracks.

UC1-Demo1 has generated two components:

- VERSAL (VCK190) inference component
- Report generation component

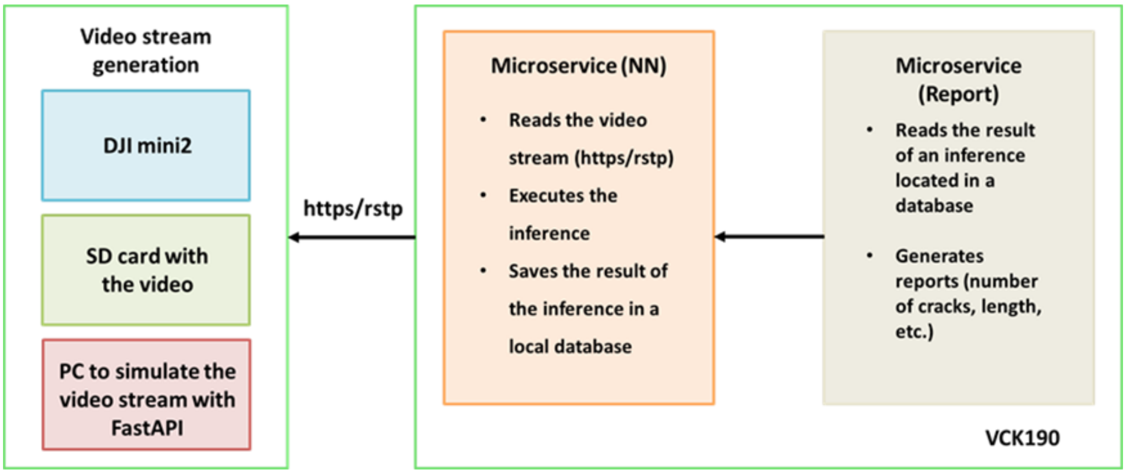

Figure 8: UC1-Demo1 components architecture

4.1.4 Validation & Assessment of KPIs

The objective of **UC1 Demonstrator 1** is the detection of cracks in concrete structures. This project will replace the current visual inspection methods that require costly auxiliary machinery, traffic interruptions and can be a source of accidents.

In order to demonstrate the cognitive awareness component, the crack perception ability of a technical expert will be compared with the performance of the AI algorithm. The KPI "number of cracks detected" will be used to evaluate this component.

Another capability that will be measured is the algorithm's ability to differentiate cracks from other imperfections such as graffiti, paint chipping, aggregate nests, etc. If the algorithm is able to detect cracks with high accuracy on different surfaces and structures, it will be considered a successful system. This "imitation" of human cognitive function will be achieved by retraining the model and by using our image augmentation module, which overlaps different textures with the original images. One way to measure this component is by using the KPI "average performance difference (number of cracks detected) between the technical expert and the algorithm by inspection". Notice that the mean performance should increase with our system, and there must be a minimum accepted performance (compared with the performance of a technical expert).

Finally, the main objective of the system is to reduce the number of accidents inherent to "traditional" inspection methods. Thus, the KPI used to measure this component will be the "reduction of the exposure time of the site worker" during the work activity on field.

FRACTAL feature Cognitive Awareness	
UC1 contributes to the development of AI-based COGNITIVE AWARENESS feature to make the FRACTAL edge node aware of its environment by identifying the surrounding.	
Technical KPIs	• (UC1 Demonstrator 1) Percentage of the number of cracks detected by the algorithm with respect to those detected by an expert **> 95%**
Business KPIs	• (UC1 Demonstrator 1) **Reduction of the exposure time** of the site worker **>50%**

Table 2: KPIs on the COGNITIVE AWARENESS feature (UC 1).

4.1.5 **Path to exploitation**

This solution will make it possible to offer an advanced inspection service to monitor the status of the structures both in construction sites during works and during maintenance phase. In this case, the target customers will be linear infrastructure concession companies, administrations that manage these terrestrial communication lines or consulting firms specialized in structural pathologies.

Retraining the algorithm through new images of structures generated by experts with active, inactive cracks and other flaws would improve the solution, providing much more valuable information in operation and maintenance stages of the structures, providing greater safety for both operators and users in operation.

4.2 Demonstrator II – UAV supervision of critical structures

4.2.1 **Addressed FRACTAL features**

4.2.1.1 FRACTAL feature Cognitive Awareness

The cognitive awareness capabilities will be enabled by a sensor network deployed over the machinery and wearable devices. These sensors will collect data about the interactions between the construction workers and the machinery, whenever an approach that can be considered hazardous happens between them. Notice that these sensors never collect personal information from the workers nor interact with them in any way (wearable sensors do not trace the positions of the workers out of alarm areas, do not collect information on how much time the worker has been on each location), so no GDPR rules are broken. This is a relevant aspect as data obtained from this sensor network is anonymized by default (the sensor ID is the only information present in the data, but not the worker’s personal information), so in case a cyber-security leak happens, no personal data will be exposed.

4.2.1.2 FRACTAL feature Cloud Communication / Connectivity

Connectivity is part of the FRACTAL Feature Fractality.

The ML models must provide a quick response, and having real-time inferences is a must, because the safety of workers will be improved if a fast response is provided to forecasted hazardous events. These real-time capabilities are achieved by having a sensor network deployed at the edge, which collects data from the scenario, positions of workers and machinery, and sends alarms to the edge controller whenever the workers approach a machine from a dangerous area being monitored. These data can then be processed directly at the edge without a strong dependency to the cloud, which results in faster responses.

4.2.2 **Building block integration and verification**

Through the deployment of a network of personal wireless sensors in the workforce and individual devices on the machinery, a safe environment will be created on the construction site. The devices allow two safety radii to be established depending on the proximity between the workers and the machinery, so that a vibration and alarm is emitted on both devices in the event of a dangerous approach. A medium danger radius is defined, where workers must be aware of the danger and act accordingly, and another high danger radius where the machinery must stop instantly in the event of an imminent accident. In addition, the machinery vehicles are equipped with GPS positioning so that their movements within the site can be recorded.

The devices chosen were those provided by the Linde Safety Guard system from LINDE, whose functional description fits perfectly with the requirements established by the UC1-Dem2: "the Linde Safety Guard is a wireless assistance system for protecting people and objects in defined danger areas in industrial environments. The system wirelessly measures the distance between the component mounted on the industrial truck, components at fixed positions in the working area and the mobile components that individuals carry on their person. In this way, the assistance system can effectively use LED displays, warning sounds and vibrations to warn people of danger and help to avoid potential collisions with industrial trucks. For an early and effective warning, the Safety Guard display unit has two warning zones. The extended area can be configured in such a way that it covers a cone-shaped area to the front and rear. Within this area, the device displays the direction towards people with a mobile warning unit. The immediate vicinity covers a circular area immediately around the industrial truck. The size of the areas can be configured and must be adapted to the work environment before initial commissioning".

The information collected is gathered and processed through data aggregation techniques, then fed to the corresponding ML models that will make their respective predictions. The outcoming data from the models and the processed data is represented through dashboards with information on alarms and machinery positioning, among other information, which will be used for a posteriori analysis to improve health and safety on site, and the generation of predictions of the most probable alarms, which will contribute to take quick measures.

The following components have been integrated

feature	building block / component	layer
Cognitive Awareness	WP4T44-05 – IoT Gateway	application
	WP5T54-01-01 - MLBuffet	service-orchestration
	WP5T54-01-01 – Training moduel for MLBuffet	service-orchestration
	WP5T54-02-01 – Docker Swarm	service-orchestration
	WP6T54-02-02 - Kubernetes-based Container Orchestrators for the Edge	service-orchestration
Cloud communication / Connectivity	WP4T44-05 – IoT Gateway	application
	WP5T54-03 – MLOps toolchain	node layer
	WP6T61-01 – Edge ML API	application
	WP6T62-01 – Data Ingestion	application

Table 3: component Id, name and abstraction layer for the second demonstrator (UC1).

4.2.3 Demonstrator setup

Demonstrator 2 is being developed in the HERNANI-ASTIGARRAGA high-speed railway project, which will link the Basque Country with Madrid. As part of this project, various concrete structures are being built. For this purpose, earthworks are being carried out where the machinery, together with the workforce, are working together on the daily tasks.

Figure 9: Demonstrator 2 deals with railways.

Figure 1 - UC1 - Demonstrator 2 scenario

In this ecosystem, a network of sensors will be deployed to collect information on possible dangerous proximity alerts through wearable devices.

This sensor network consists of three different devices:

This sensor network consists of three different devices:

- Device for machinery

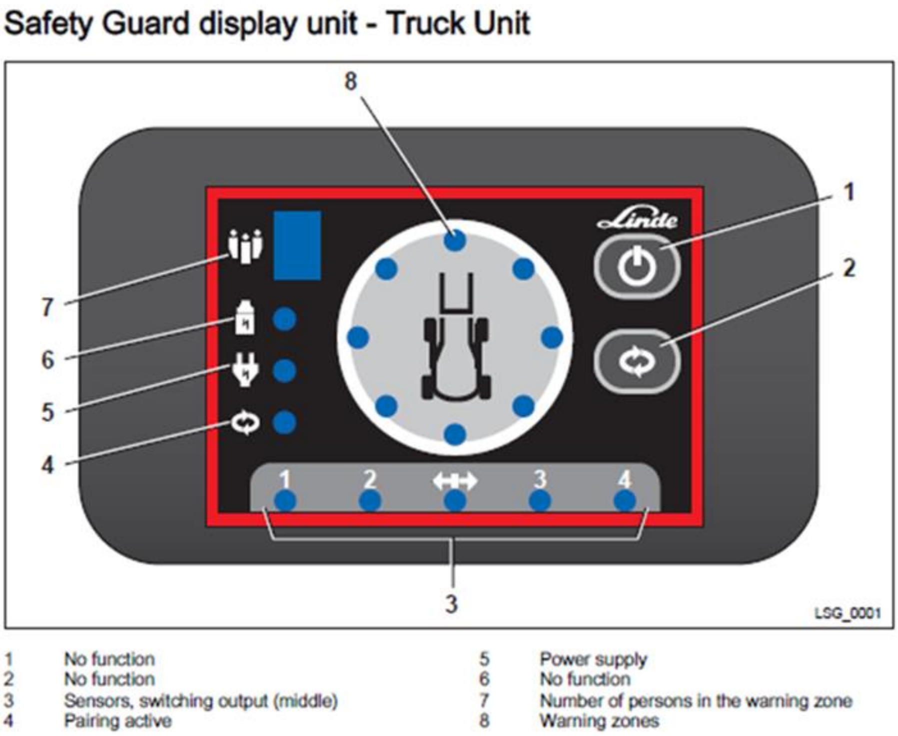

Figure 10: Safety guard display unit.

- Machinery GPS

- Workforce devices

Figure 11: Safety guard warning unit.

For the deployment of the pilot project, various procedures have been carried out with the construction company (Sacyr, S.A.) to ensure the success of the installation and subsequent monitoring of the technology. In relation to the machinery, it has been necessary to install three clamps for power in the machinery to be used. There are three types of machines, two of which will be equipped with sensors for the pilot project:

Vehicle	Self-propelled crane
Brand	LIEBHERR
Model	LTM1050/1
Location	Central viaduct of Hernani
Use	Steel framework elevation in the viaduct deck: Steel bars with different length (up to 12 m)

Figure 12: Self-propelled crane for steel framework elevation.

Vehicle	Self-propelled crane
Brand	LIEBHERR
Model	LTM1060/2
Location	Central viaduct of Hernani
Use	Prefabricated elements elevation

Figure 13: Self-propelled crane for prefabricated elements elevation.

Vehicle	Backhoe excavator
Brand	CASE
Model	WX 185
Location	Diversion 3 excavation
Use	Land movement (excavation, load, widespread land, landfill, etc.)

Figure 14: Backhoe excavator.

Work has been carried out to prepare the machinery for the next connection of the devices by means of wiring.

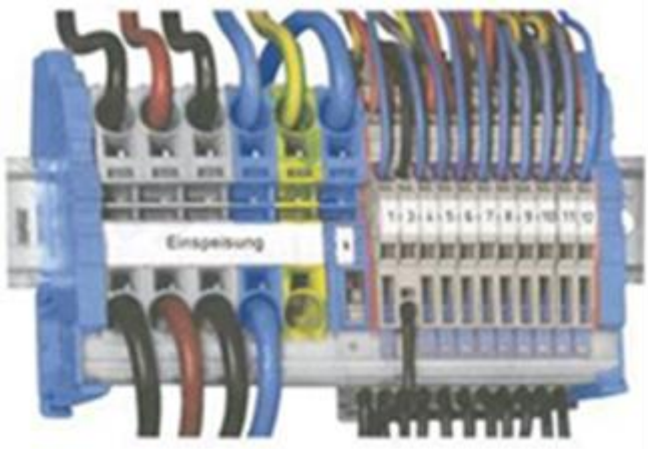

Figure 15: Wiring to prepare machinery for connection of the devices.

UC1-Demo 2 has generated three building blocks for AI components:

- **Alert predictor**: the model determines if the relative positions of workers and machinery constitute a hazardous situation.

- **Alert classifier**: It tells the user the nature of the alarm (machine-machine, worker-machine, etc.).
- **Anomaly detector**: It detects what events in a time-series formatted dataset have special features. This model tells when the algorithm failed or succeeded in its predictions.

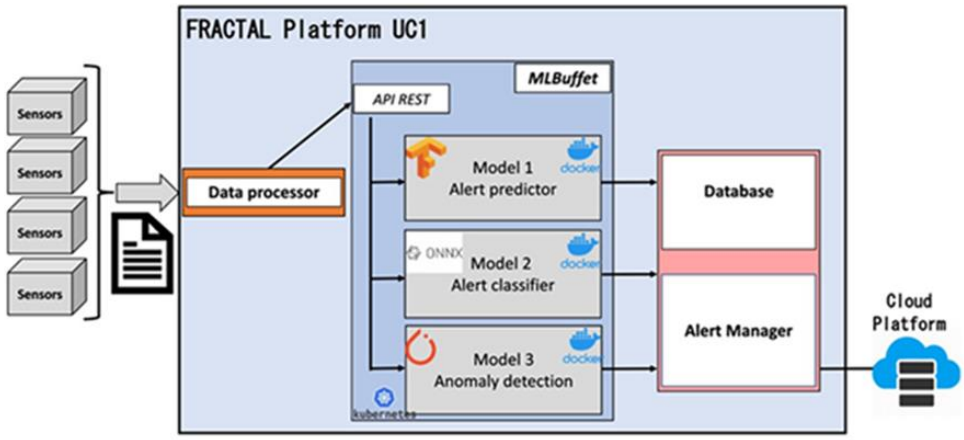

Figure 16: UC1-Demo2 data flow architecture

This architecture shows how the different AI Building blocks interact with each other. Firstly, the information from the LINDE sensor network goes to the Data processor, which is a system that formats the data accordingly to the models' expected input, and splits the different entries into HTTP requests to be sent to the MLBuffet's REST API holding the models. The models consist of Neural Networks trained with the Tensorflow v2 library from the historical gathered data from the LINDE sensor network. Once trained, the models can be retrained to improve their metrics and performance once enough data is available.

This REST API collects the inputs and feeds them into the different models, providing the results of the inference into a Database and the Alert Manager simultaneously. The Database will keep the data cached for a period of time, to make sure that the data are backed-up and stored for historical dataset creation. The Alert Manager is configured with a set of rules based on the models' outputs and will generate alerts for the platform operator to take the appropriate actions.

These actions vary depending on the nature of the alerts, which could be high risk of hazardous situations predicted, type of alert predicted when a risk is encountered, or anomalous behavior being detected. Finally, the gathered data, together with the predictions and the LINDE raw data are collected and sent to an external Cloud Platform, which will only serve as data history, and post-processing of these data for further model refinement or re-training.

Notice that the external Cloud should be differentiated from the Fractal Cloud Platform, as the latter serves as a support for the Edge AI operations, but in this case, no support is needed from external Clouds, as the Fractal Platform designed for UC1-Demo2 should be independent and should be able to work in environments with no internet connection.

This technological stack will be installed and deployed in a containerized approach, into the VERSAL board. Other Xilinx boards can be used as long as they support Linux distributions (Ubuntu or PetaLinux custom distributions) and have an ARM processor architecture.

Other technical limitations include the quality of the data to build the ML models and the data set limitations. Human behavior is intrinsically unpredictable and obtaining behavioral patterns requires a large amount of data and long data collection times (1-2 years). This has a direct effect on the quality of the generated AI models, which will have a poor performance at first but will see their results enhanced while new data are being collected.

The collected data has the following structure:

gps_data: These data show the GPS information about each sensor deployed on the construction scenario. Note that the data are fully anonymized and normalized so no tracking of any worker can be done and no personal information can be obtained from this data.

	uid	device_mac	timestamp	latitude	longitude	acceleration	loading_distance
0	89390114	DCA632E35686	1663678443	NaN	NaN	1.052127	87.0
1	89390115	DCA632E35686	1663678444	NaN	NaN	1.052127	79.0
2	89390116	DCA632E35686	1663678445	NaN	NaN	1.048406	86.0
3	89390117	DCA632E35686	1663678446	NaN	NaN	1.048406	87.0
4	89390118	DCA632E35686	1663678447	NaN	NaN	1.048665	81.0
5	89390119	DCA632E35686	1663678448	NaN	NaN	1.048665	81.0
6	89390120	DCA632E35686	1663678449	NaN	NaN	1.048665	95.0
7	89390121	DCA632E35686	1663678450	NaN	NaN	1.048665	86.0
8	89390122	DCA632E35686	1663678451	NaN	NaN	1.048665	83.0

Figure 17: Fully anonymized GPS data.

Sensor interaction data: These data show the interactions between sensors, providing information about the type of sensors interacting, the entry and exit zones from the alarm areas, duration of the interactions and red-zone alarms. With this dataset, Recurrent Neural Networks can be built that predict future alarms and hazardous interactions based on time-series oriented data.

	DestMac	DestType	InRedZone	EntryZone	ExitZone	StartTime	EndTime	PositionX	PositionY	SrcMac	IRDist
0	10002D5C10205FC9	BEEPER	False	1.0	1.0	1.663683e+12	1.663683e+12	0.000000	0.000000	10002D5C10205FC9	0.0
1	10002D5C10205FC9	BEEPER	False	1.0	1.0	1.663683e+12	1.663683e+12	43.279251	1.957112	DCA632E35686	290.0
2	10002D5C10205FC9	BEEPER	False	14.0	14.0	1.663683e+12	1.663683e+12	0.000000	0.000000	DCA632E35686	0.0
3	10002D5C10205FC9	BEEPER	False	14.0	14.0	1.663683e+12	1.663683e+12	0.000000	0.000000	10002D5C10205FC9	0.0
4	10002D5C10205FC9	BEEPER	False	1.0	1.0	1.663683e+12	1.663683e+12	43.279196	1.957134	DCA632E35686	290.0
5	10002D5C10205FC9	BEEPER	False	1.0	1.0	1.663683e+12	1.663683e+12	0.000000	0.000000	10002D5C10205FC9	0.0
6	10002D5C10205FC9	BEEPER	False	14.0	14.0	1.663683e+12	1.663683e+12	0.000000	0.000000	DCA632E35686	0.0
7	10002D5C10205FC9	BEEPER	False	14.0	14.0	1.663683e+12	1.663683e+12	0.000000	0.000000	10002D5C10205FC9	0.0
8	10002D5C10205FC9	BEEPER	False	1.0	1.0	1.663683e+12	1.663683e+12	43.279107	1.957078	DCA632E35686	290.0
9	10002D5C10205FC9	BEEPER	False	1.0	1.0	1.663683e+12	1.663683e+12	0.000000	0.000000	10002D5C10205FC9	0.0

Figure 18: Sensor interaction data.

The collected data is presented on a dashboard for the safety control during the execution of the work, as well as the monitoring of the movements of the machinery. In this way it is possible to observe by hours, days and devices those moments in which an alarm has occurred since the operators have been too close to the moving machinery.

Figure 19: Dashboard for safety control during the execution of work.

4.2.4 Validation & Assessment of KPIs

The main objective of the UC1 Demonstrator 2 is to improve occupational safety and health in civil works environment. Cognitive awareness is a novel feature in these kind of construction scenarios. Although Machine Learning models were available and have been applied before to construction sites, their applicability was limited to predictive maintenance works, anomalous functioning of machines, or other kind of machine-related aspects. Being able to cognitively adapt to the environment and the present element in the scenario opens a gate to new models being deployed.

The KPI for measuring the impact of cognitive awareness capabilities in the construction scenario would be the monitoring of the IoT platform data model's performance in terms of accuracy and reliability.

An alarm prediction model that is trained by just knowing the overall number of people and machinery on the working site should be outperformed by a model that actively knows how many people and machines are involved in each alarm event. The model should work better if the number of people on site can be detected automatically. A comparison between the model's performance before and after enabling cognitive awareness would be enough to determine whether cognitive awareness is a relevant aspect in the model's accuracy.

FRACTAL feature Cognitive Awareness	
UC1 contributes to the development of AI-based cognitive awareness feature to make the FRACTAL edge node aware of its environment by identifying the surrounding.	
Technical KPIs	• (UC1 Demonstrator 2) The model works better when the number of people in the scenario can be detected automatically (**Improved quality with people detection**)
Business KPIs	• (UC1 Demonstrator 2) Make the system capable of determining how many people there are on stage at any given moment and configure itself based on this (**People detection capability**)

In addition, for the functional validation of the results and the technology developed, surveys have been prepared for the operators and drivers who have worked during the development of the pilot in the field. In this way, we will be able to see the acceptance of the system during the work, as well as to improve those elements that may have hindered or created discomfort during the working day.

CUESTIONARIO **OPERARIO** PILOTO PROYECTO FRACTAL

FECHA: ____________________

¿Cuántos días llevó el dispositivo?	☐ De 0 a 10	☐ De 10 a 20	☐ Más de 20
¿Con qué máquina trabajó llevando el dispositivo?	☐ Giratoria	☐ Camión-grúa	☐ Con
¿Le resultó cómodo el llevarlo encima?	☐ S	☐ N	
¿Le resultó molesto el llevarlo? (Por las alarmas, las vibraciones, etc.)	☐ S	☐ N	
¿Tuvo la tentación de apagarlo?	☐ S	☐ N	
¿Lo hizo? (La encuesta es anónima)	☐ S	☐ N	
La batería del dispositivo, ¿aguantó toda la jornada	☐ S	☐ N	
¿La alarma sonora se oía bien?	☐ S	☐ N	☐ Depend
Si en la última pregunta respondió "depende", ¿qué causas cree que fueron el motivo de no escucharla? (Marque las opciones que desee)	☐ Ruido de la máquina que llevaba el ☐ Ruido de otras máquinas cercanas ☐ El dipositivo se oía poco, incluso al máximo ☐ Ruido de la máquina que llevaba el ☐ Otros ____________________		
¿La vibración se notaba con facilidad?	☐ S	☐ N	
¿Cree que un sistema de este tipo es útil?	☐ S	☐ N	
Si en la última pregunta respondió "no", ¿por qué causas cree que es así? (Marque las opciones que desee)	☐ Ruido de la propia máquina que llevaba los ☐ Ruido de otras máquinas cercanas ☐ El dipositivo se oía poco, incluso al máximo ☐ Ruido del propio trabajo que estábamos ☐ Es suficiente conque sólamente suene en la ☐ Otros ____________________		
¿Con qué otra maquinaria cree que el sistema sería más útil? (Marque las opciones que desee)	☐ Maquinaria de vía ☐ Maquinaria de asfaltado ☐ Grúas para colocación de prefabricados (vigas, tubos, ☐ Otros ____________________		

Figure 20: Survey to involve operators and drivers.

4.2.5 **Path to exploitation**

Safety is one of the most important elements on construction sites, a sector traditionally linked to high accident rates. Moreover, the lack of digitisation and limited use of health and safety technologies means that these levels are not being reduced as quickly as they should be. The system developed within FRACTAL will not only reduce potentially dangerous situations in real time, but will also provide key information to understand why these situations have been triggered.

The system that will be developed within the project has a limited amount of data for the months where the devices have been deployed on site, so that in future developments the volume of data collected will allow the solution to be improved, especially the alarm prediction system powered by the fractal node.

As for the potential market for the solution, it is clearly focused on development in construction environments, so construction companies would benefit the most from the implementation of this system. In addition, the solution can serve as a security control for administrations and operators, helping in the drafting and justification of the security measures adopted during the course of construction.

5 UC2 – AI-based controls for thermal management

5.1 Addressed FRACTAL features

As illustrated in Deliverable D7.2 – considering the state of the practice - thermal management of Battery Electric Vehicles (BEVs) is performed using a model predictive control (MPC) where the model is designed from first principles. MPC tends to fail or underperform when the vehicle is driven in scenarios outside of the boundary conditions that have been assumed during the design of the control strategies. Ideally the control strategy needs to be capable of learning from historic behavior (varying environmental temperature, different driving style, unseen maneuvers etc.) and adapt accordingly. In that sense the use case demonstrated two major features of FRACTAL, namely adaptability and cloud communication.

This use case on reinforcement-learning (RL) based controls disrupts a long-standing paradigm in the automotive industry: Following the automotive V-model, there is a strict separation between the design, implementation and testing phases and the in-use of a vehicle function. To leverage historic data from the in-use phase the automotive industry needs to embed the DevOps and ML-Ops paradigm to the automotive V-model. With the focus on adaptability this use case contributes to achieving this goal as it demonstrates how to implement the ML-Ops loop within a dedicated automotive use case on EVs. It furthermore assesses specific KPIs on adaptability, cloud communication and openness within this prototypical setup.

5.1.1 FRACTAL feature Adaptability

adaptability is the key feature of UC2. UC2 instantiates adaptive behavior in terms of a loop starting with training first version of a control strategy within the FRACTAL cloud environment using ML framework coupled with high-fidelity simulation of the thermal behavior of the cabin and the powertrain. That model is deployed to a FRACTAL edge node. While performing the inference step at the edge we collect new sensor data. This data is uploaded to the FRACTAL cloud node where it is used to improve the control strategy to retrain the model.

5.1.2 FRACTAL feature Cloud Communication

After training, RL-based control strategy (RL-model) is exported to the Open Neural Network Exchange (ONNX) format. It is tested on fixed drive cycles and as soon as this the RL-based control strategy performs better than the old version, the model repository is updated and Docker image with the improved RL-based strategy is deployed to the FRACTAL edge node running on the VERSAL board.

While performing the inference at the edge data is recorded and later uploaded to S3 compatible data storage[4]. That data closes the loop, allowing the RL-framework running on the FRACTAL cloud node to iterate to the next version of the model.

5.1.3 FRACTAL feature Openness

UC2 make use of a number of open source components and contributed to improving open source solutions.

5.2 Building block integration and verification

The training of the control strategy is done at the cloud node and follows the paradigm of reinforcement learning. There are two versions of training – online and offline training. Online training uses training data generated by the simulation. The trained agent interacts with a simulated car using predefined drive cycles. Offline training is done using data recorded from actual runs on the edge node. The employed ML model is a feed-forward neural network implemented using the PyTorch Python library[5]. The neural network is trained using dedicated algorithms tailored to the reinforcement learning paradigm, e.g., Proximal Policy Optimization[6] (PPO) or Advantage Actor-Critic[7] (A3C) for online training and MARWIL[8] for offline training. The following table lists the employed components at the application, orchestration at the node layer for the features adaptability and CONNECTIVITY.

Systemic property (FRACTAL feature)	Component ID	Component name	Layer
Adaptability	WP3T34-03	VERSAL Model deployment layer	application
	WP6T61-03	Data analysis	application
	WP5T52-04-01	Model version control	application
	WP5T52-04-03	S3 compatible data storage	application

[4] https://aws.amazon.com/s3/
[5] https://pytorch.org/
[6] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347.
[7] Mnih, Volodymyr, et al. "Asynchronous methods for deep reinforcement learning." International conference on machine learning. PMLR, 2016.
[8] Qing Wang, Jiechao Xiong, Lei Han, Peng Sun, Han Liu, and Tong Zhang. 2018. Exponentially weighted imitation learning for batched historical data. In Proceedings of the 32nd International Conference on Neural Information Processing Systems (NIPS'18). Curran Associates Inc., Red Hook, NY, USA, 6291–6300.

	WP5T52-04-05	Datasets version control	application
	WP5T52-04-07	Images repository	application
	WP5T52-04-08	Model repository	application
	WP5T52-06-01	Model preparation for Fractal Edge (VERSAL Xilinx Vitis AI)	application
Cloud Communication / Connectivity	WP5T52-06-02	ML pipeline connection to model repository	application
	WP5T52-01-01	Data Ingestion Service	application
	WP5T52-02-01	Raw data Object storage service	application
	WP5T52-07-01	Kubernetes-based cloud platform container orchestrator	service-orchestration
	WP5T52-07-02	Cloud container orchestrator services access	service-orchestration
	WP6T61-02	Edge API	service-orchestration
	WP3T36-02	Load Balancing Module	service-orchestration

Table 4: component ID, component name and abstraction layer used to implement the systemic properties adaptability and CONNECTIVITY.

5.3 Demonstrator setup

To achieve adaptive behavior, we employ the FRACTAL cloud as well as the FRACTAL edge platform. On the cloud platform, we use reinforcement learning (RL) to implement a training and (re-training) loop. To achieve high accuracy in RL, we employ a high-fidelity vehicle model alongside with dedicated driving cycles to train the RL agent. Once we have achieved an adequate model (see model KPIs) we deploy the model for inference to the FACTAL edge platform. Aiming at TRL 5, the edge platform acts as a representation of the real vehicle. Figure 1 describes the overall setup of the demonstrator. Details of the implementation are given in Deliverable D7.2.

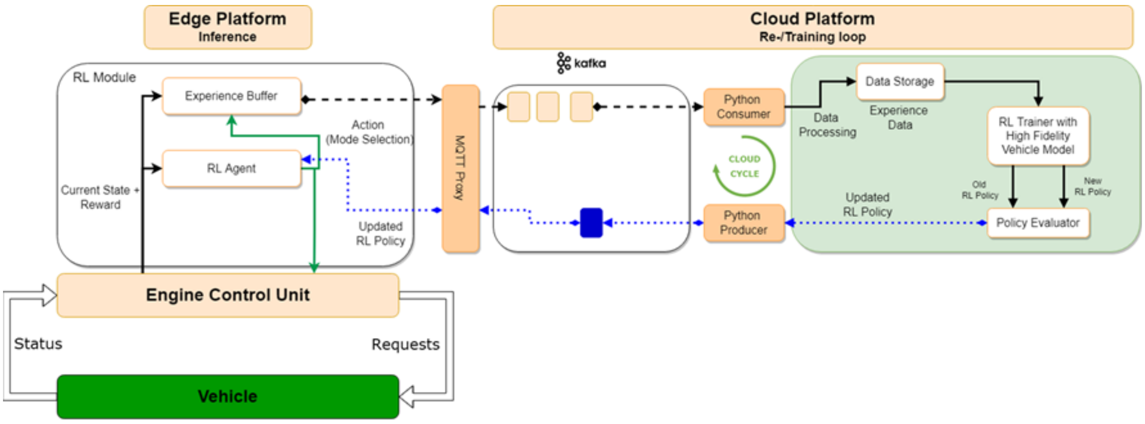

Figure 21: The demonstrator setup for UC2 on RL-based controls, cloud- and edge platform alongside with the high-level architecture of the application.

For initial training in the simulation, we use 6 different drive cycles stored on LakeFS data repository. Each drive cycle consists of 3 different back and forth car rides:

- Ride #1: Driving range on city/suburban (daily commute, taxi)
- Ride #2: Long highway trip and mountain drive to ski station
- Ride #3: Long mountain trip, with high accumulated positive and negative elevation

All 6 drive cycles are based on real world road data and have been augmented with simulated traffic and weather conditions. The first two rides are based on the roads around Paris (ride #1 #2), and third ride (ride #3) is in the Austrian/German Alps.

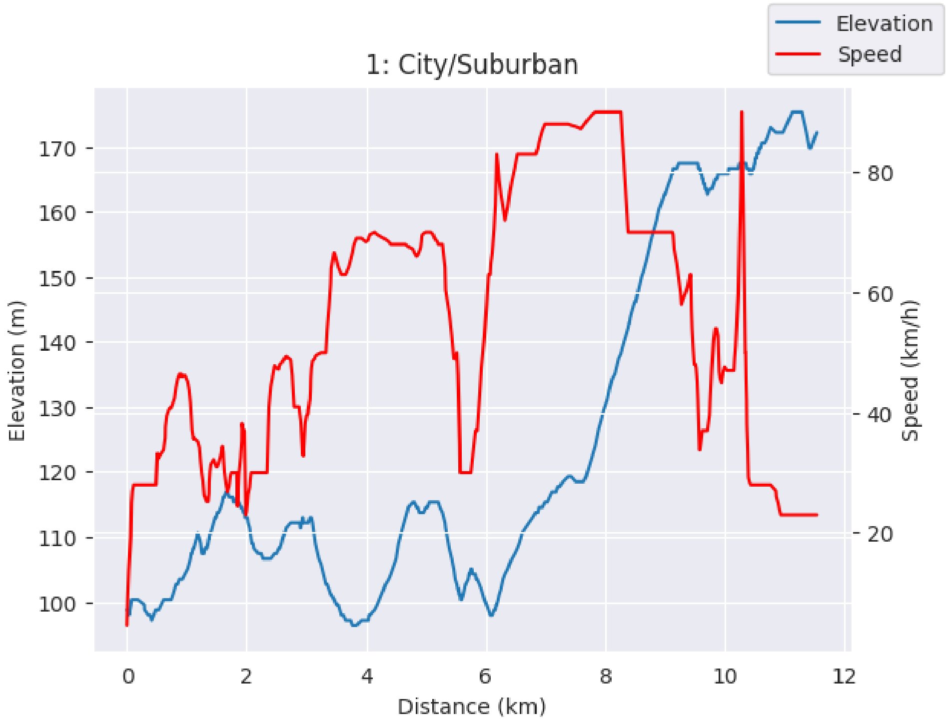

Figure 22: Excerpt of elevation and vehicle speed for the city/suburban drive cycle. The data has been used for training the ML-model used for implmenting and subsequent assessment of the adaptive control strategy.

Distances covered per drive cycle range from 11.3km in city/urban settings to 626km in long mountain drive. The maximum obtained speeds are 90km/h in urban settings and almost 130km/h in other two drive scenarios.

To achieve bigger variability in training data, we have chosen to not use the original drive-cycle weather conditions, but to define multiple versions of conditions for the same drive cycle. In our training experiments, we are cycling within a temperature range from -7°C to 10°C. Also, since we have 6 drive cycles and different length could bias trained RL agents, we've chosen not to use whole drive cycle every time but allow slicing part of drive cycle which duration is 110 minutes.

The vehicle is modeled by high fidelity simulation compiled to Functional Mockup Unit (FMU). There are 20 inputs to our current FMU: 11 input variables set upon initializing FMU (e.g., the initial cabin temperature or battery state of charge). Other 9 are readouts from the drive cycle (e.g., vehicle speed) as well as heating mode decision provided by RL agent and reset signal for reinitializing the simulation. During FRACTAL JU engineers have designed 8 different heating modes – each mode describing a distinct combination of heating and cooling systems working to keep the cabin temperature at the desired level.

FMU is outputting 27 signals, most of them various temperatures and pressures in car systems, alongside with the state of charge (SoC). Internal timestep provisioned to the FMU is 0.1 seconds, which means the FMU returns control to RL-framework running as soon as 0.1 seconds of simulation time have progressed.

FMU is executed by FMPy Python library wrapped in OpenAI's gym environment (see Table 4, open-source components). Gym environment takes care of executing atomic FMU timesteps and returns control to the agent when it is expected to make a decision. While this time window does not need to be regular, we have decided to make it a constant but configurable interval. We defined this agent step to be 60 seconds in our training process.

Agent is expected to keep car's systems in the safe and efficient zone. Desired zones are defined by their soft and hard limits. Soft limits refer to the optimal or preferred range of values for the readouts from the car, but the system may still function if the values fall outside of this range. Hard limits refer to the minimum and maximum values that the readouts must not exceed in order for the system to function properly. Episode execution stops if either whole drive cycle slice was covered, or if some of internal car parameters have breached its hard limit, which would place the simulated car in the unsafe zone.

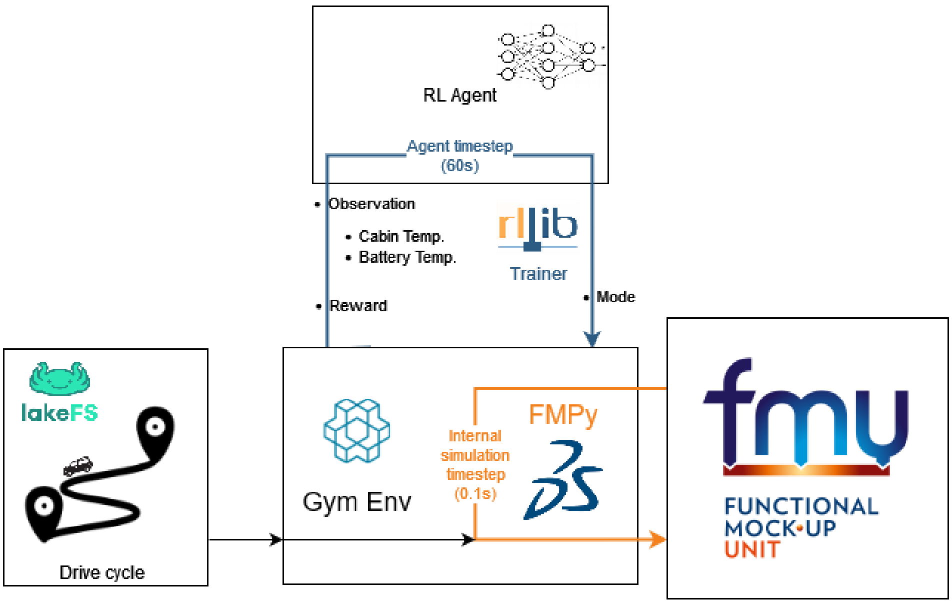

Figure 23: Schematic use of the application-level components on top of the FRACTAL cloud node.

Gym environment, besides collecting observations, calculates the reward for the last action executed by agent. RL training is very sensitive to both reward function and an agent's search space. Thus, we've created an easy way to construct the reward function and determine observation space. Observation space can easily be constructed by either including or excluding a set of FMPY and drive cycle outputs.

A key element in implementing RL-strategy is the reward function used in RL. I our case the reward function is a linear combination of chosen normalized observation values and additional derived metrics, e.g., cabin temperature and deviation of cabin temperature from target temperature. Also, breaching a soft limit is punished by linearly increasing the punishment until the value reaches a hard limit. Breaching a hard limit can be punished with additional punishment. After a couple of iterations, we have come up with the following limits settings for our training process:

Limit / Measurement	Battery temperature [°C]	Powertrain temperature [°C]	State of Charge [1]
Factor	1	1	0
Lower hard	-30	-30	0
Lower soft	0	-10	0.1
Upper soft	50	50	0.9
Upper hard	60	70	1
Extra penalty for hard breach	100	100	0

Table 5: Breaking of lower and upper soft- and hard limits for battery temperature, powertrain temperature and SoC contributes to agent's knowledge through the reward function. Extra penalty for hard breach is an additional punishment for breaching a hard limit.

State of Charge (SoC) refers to the usable battery capacity. Usable battery capacity is limited between 10% and 90% of the actual (gross) battery capacity to preserve battery health. Factor 0 for normalized state of charge limit means we're stopping a simulation after hard limit is reached, but we are not punishing neither breaching a soft nor reaching a hard limit. Since battery and powertrain temperature both have factor 1, breaching those two limits is equally important in the final reward function. Besides punishing breaching battery and powertrain temperature limits, we are punishing a drop in SoC, mean deviation from target temperature during agent step, pressure in the compressor as well as pressure in the chiller.

Our online training stops after 2 million agent steps. Our agent is evaluated against other agents and the best agent is deployed on the FRACTAL edge node. For online training we use PPO algorithm.

Offline training does not require hi-fidelity simulation, but it is being done on trajectories recorded on LakeFS[9] at the edge. In RL, trajectory is a sequence of observations, actions and rewards agent has seen, chosen and claimed during its run. For offline training we use MARWILL algorithm.

[9] https://lakefs.io/

Reinforcement learning is being run on Ray cluster deployed on Kubernetes via Kuberay[10] operator. The head size limit, worker size limits and the overall number of workers are defined in a Helm chart. For deploying Ray cluster, the auto-scaler takes care of creating and destroying worker nodes to match the workload needs.

For our evaluation we have used the following three acceptance criteria:

- average SoC usage per 100km as measured on testing cycles
- time required for cabin temperature to approach 1°C zone around the desired temperature
- and distance covered by car in simulated test run until it drained the battery.

Agents are scored on all 3 criteria by calculating each agent's position in normalized distribution of all agents. Scores are weighted: 60%, 20% and 20% respectively, and the best one is chosen for deployment to the edge.

5.4 Validation & Assessment of KPIs

Performance of RL agents is measured in terms of optimal solution, often provided by the dynamic programming technique. Key criteria for thermal management use case are (1) decreasing SoC, (2) reaching the desired cabin temperature as fast as possible and (3) keeping the temperature fixed.

Thus, we define following measurements:

- **dSoC/100km** - State of charge usage, normalized per 100km.
- **Rise time/°C** – We define rise time as time passed until cabin temperature reaches 1°C stripe around the desired temperature. Rise time is normalized by the difference of starting and target cabin temperature.
- **Temperature keep rate** – percentage of time spent in 1°C tolerance band around the target temperature. It is measured only after the cabin temperature reaches the acceptable range for the first time.
- **Mode changes per hour** – number of mode changes per hour.

[10] https://ray-project.github.io/kuberay/components/operator/

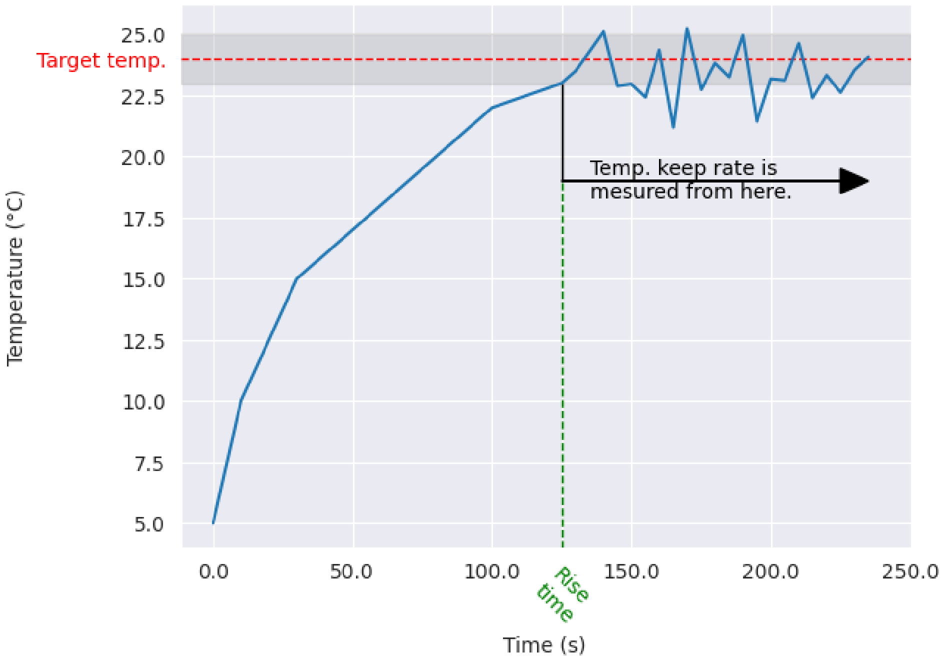

Figure 24: Illustration of key metrics to capture the dynamics of the adaptive control strategy.

dSoc/100km is not translatable across different testing scenarios and special care must be taken that all agents are tested in the same conditions. E.g., dSoc/100km measurement will be biased in the favor of routes with more descent due to regenerative breaking.

<table>
<tr><th colspan="2">FRACTAL feature adaptability</th></tr>
<tr><td colspan="2">UC2 contributes to the adaptability feature by design, implementation and empirical assessment of a RL-enabled control strategy and enables a controlled system to continuously evolve depending on perceived context.</td></tr>
<tr><td>Technical KPIs</td><td>• The degree to which the control reaches its goals, deviation measured from Dynamic Programming solution should be on the scale of 5% or less for the reinforcement learning based solution. This is measured by dSoC/100km and rise time.
○ Validation results: see Table 7 for refined metrics
• The data-driven control strategy should be better on average when compared to the reference strategy (i.e., the model/rule-based design of the control) for unseen drive cycle, measured by dSoC/100km on the cycle.
○ Validation results: see Table 7 for refined metrics</td></tr>
<tr><td>Business KPIs</td><td>• Reduction of initial calibration effort – at least 25% of PM.
○ Evaluation is ongoing, will be assessed in D7.3(2)
• Allows constant updating of calibrated models, adjusting them to structural changes specific to each car during its lifetime – infeasible in the MPC approach.
○ Validation results: ML-Ops loop demonstrated
• Responsiveness for unseen scenarios
○ Evaluation is ongoing, will be assessed in D7.3(2)</td></tr>
</table>

Table 6: Technical and business KPIs for the adaptability feature as sketched in Deliverable D7.1 Building block verification plan.

Given the sketched demonstrator setup, we obtained the following KPIs:

Technical/Business KPI	RL agent	Baseline solution (reference strategy)	Optimal solution (dynamic programming)
dSoC/100km	21.31	26.2	to be assessed in D7.3(2)
Rise time/°C	38.64	30.01	to be assessed in D7.3(2)
temp. keep rate	100%	100%	to be assessed in D7.3(2)
mode changes per hour	to be assessed in D7.3(2)	to be assessed in D7.3(2)	to be assessed in D7.3(2)

Table 7: Empirical assessment of technical KPIs regarding adaptability. The RL-agent outperforms the baseline strategy at acceptable rise time and manages to keep the cabin temperature in the required tolerance band. Agent was evaluated on drive cycles 1,2 and 3 (rides #1 and #2).

The second FRACTAL feature addressed with UC is connectivity to the cloud node.

FRACTAL feature Cloud communication	
UC2 contributes to the FRACTAL feature CLOUD communication as it uses Apache Kafka for data routing when it to comes to pre-training of the ML model.	

Technical KPIs	• Number of pushed messages for pre-training / frequency of messages: **up to 100 messages per day per IoT device**.
	  ○ *Validation result: our experiment needed 72MB of data within 8 hours, see argumentation below*
	• Size of batches – **> 1MB**
	  ○ *Validation result: Considered adequate for our experiments, see argumentation below*

Business KPIs	• **> 99%** of information sent to Kafka is retained and consumed. ○ To be assessed in D7.3(2)

Table 8: UC allows for upgrading the RL-based control strategies by re-training the models on the FRACTAL cloud node. The table (also see D7.1) specifies specific KPIs on CONNECTIVITY as perceived in UC2.

Depending on the chosen the observation space, each agent step records 10-30 values. Datatype double in Python is 8 bytes large, thus each agent step should record up to 0.25KB of data. Our current agent step (1 minute) generates up to 0.12MB of trajectory data in 8 hours ride, but finer recording of car ride – taking a snapshot each 0.1 second - generates 72MB of data in 8 hours ride. *Requiring a system to accept up to 100 batched messages daily of size over 1MB per IoT device allows us to comfortably ingest all this data.* Communication KPI requirements are calculated with the assumption that the vehicle will be used up to 8 hours per day. If that amount is exceeded, the vehicle can send to the data injection API batches of logs larger than 1MB.

While implementing UC2 we have used several open-source frameworks and open standards. As a contribution to openness, we briefly sketch the used artifacts in Table 9:

UC2: AI-based-Control	**License**
Open Standards	
Protocol Buffers (Protobuf) - cross-platform data format used to serialize structured data. (https://github.com/protocolbuffers/protobuf)	Apache License 2.0
Open-Source Software / Libraries	
Ray – Unified framework for scaling Python and AI applications. Contains scalable SoA reinforcement learning library RLlib. (https://github.com/ray-project/ray)	Apache License 2.0
KubeRay – Toolkit to run Ray applications on Kubernetes.	Apache License 2.0
OpenAI Gym – library (https://github.com/openai/gym)	The MIT License
FMPy – Python library to simulate functional mockup units – FMUs (https://github.com/CATIA-Systems/FMPy)	2-Clause BSD license

Table 9: Open standards and open-source libraries alongside with license model being used. Notably the work on UC has resulted in cleaning up several bugs both in Ray and PyArrow.

UC2 is built on Open-source technologies. Cloud training is done using Ray framework running on Kubernetes cloud deployed using KubeRay toolkit.

Ray provides a state-of-the-art machine learning framework which can seamlessly scale from run on a single computer to the cloud environment. The most important part of Ray for UC2 is RLlib, a framework for training reinforcement learning algorithms. It is exploiting Ray's scalability to implement a very robust version of all important RL algorithms used today. To run Ray on Kubernetes, it had to be deployed using helm operator based on another open source tool from Ray's ecosystem - KubeRay.

While Ray is a workhorse for UC2, development of this use-case has also reflected back on Ray: *Work on Fractal has resulted in cleaning up several bugs both in Ray and its open-source dependencies (PyArrow), as well as improving documentation for deployment of KubeRay.*

Online reinforcement learning requires training environment in which RL agent will be trained. UC2 utilizes open-source tools FMPy and OpenAI's Gym for running simulation based on FMU models. FMPy is used to run FMU model, and OpenAI Gym is used as the interface which allows RL training algorithms to interact with the simulation.

5.5 Path to exploitation

Customer acceptance on BEVs is pivotal towards the European economy. European cars attract customers around the globe. Superior design, driving dynamics, quality and functionality even allow selling for a premium price. Today the European automotive industry contributes more than € 800bn (about 7%) to the EU26 GDP. This needs to be defended against competition from USA and Asia. Customers are still reluctant to buy BEVs due to high costs. Further reducing the cost of the drivetrain is an important means to make BEVs more attractive for car buyers.

With the provided experimental demonstrator setup we disclose that the transition to RL-based control strategy is indeed applicable and has the potential for considerable productivity improvements and thus cost savings in the engineering phase of the vehicle. We further demonstrate, that closing the loop between the vehicle (FRACTAL edge node) and the back end (FRACTAL cloud node) allows for designing upgrade-able and more adaptive control strategies. Both, reducing the engineering costs due to the reduced calibration effort and extending component lifetime via improved thermal management (due to an adaptive control strategy) is key to promoting the EV vehicle market uptake. Given the numerous applications on ML-based control strategies in BEVs - e.g., 8 modes where different energy source combinations are used to heat up or cool down the cabin, powertrain and battery have been identified throughout UC implementation - AVL will follow to major paths to exploitation:

In the short term AVL will provide engineering service offerings to OEM-s: design, implementation, and validation of RL-based adaptive control strategies bundled with integration offerings into AUTOSAR Adaptive compliant EV-architectures.

In the mid-term AVL aims at integrating the proposed methodology in the form of consumable services (licenses, and via dedicated SaaS offerings) into its FASER^{TM} engineering data analytics platform thereby enabling automotive engineers (OEM-s, TIER-1-s) to overcome the barriers when it comes to designing, implementing, and testing RL-based control strategies for EVs.

6 UC3 – SmartMeter

6.1 Addressed features

The smart meter use case focuses on building a platform that can interface digital meters as well as analog meters by means of taking a picture of the meter and analyzing it. Since the device will be placed in an environment where signal reception is difficult, the device needs to support a wireless protocol that can transmit small chunks of data even in bad SNR conditions. This is achieved by interfacing a NB-IoT modem. Two other important aspects are security and low power consumption which are the main features that are demonstrated in this use case.

6.1.1 FRACTAL feature Security

The device is going to be placed near the meters at the end user where it can be accessed by several people. It is important to have the possibility to encrypt the collected data before it is stored on the device and being transmitted wirelessly. In addition, it is important that the device cannot be altered for example by manipulating its firmware. Encryption can be done on the device itself with a standard AES encryption which requires a private key stored on the device. This key must be protected which is typically done with a secure boot mechanism that protects the boot process and verifies the firmware upon boot. The secure boot process and encryption / decryption are shown in this use case.

6.1.2 FRACTAL feature Low power

To enable a battery powered mode with a multiyear battery lifetime, the smart meter use case is using the low-end Pulpissimo node that is optimized for low power consumption. To further reduce the power consumption, clock gating techniques and power domain partitioning are used. The multiyear battery lifetime is demonstrated using an example with a ML inference demo that takes a picture, analyzes it, and sends the data over the cellular network, using the NB-IoT protocol, to a server.

6.1.3 FRACTAL feature Openness

UC3 make use of a number of open source components.

6.2 Building block integration and verification

The functionality of RTL components such as WP3T32-10, WP3T32-08 and WP3T32-11 have been verified stand alone in RTL simulations (as part of WP3) with a dedicated testbench. The components have then been integrated in the hardware node (WP3T32-02). The full node has then been mapped to an FPGA where more complex applications can be run than simple tests.

Low power techniques, such clock gating, have been inserted in RTL to gate the clocks of larger blocks, and during synthesis for fine-grained clock gating. The Correct functionality of coarse-grained clock gating was verified using RTL simulations as well as functional tests on the FPGA. The power savings of clock gating, as well as power-gating of different power domains are analyzed using gate level simulations of the final design with subsequent power analysis in a digital backend implementation tool.

On the software side, the Pulpissimo platform which is running the freeRTOS operating system (WP3T32-04), has been programmed with the ML inference demonstration application (WP3T32-05) that allows to take an image and extract numbers and letters from it.

The required blocks for a secure boot process and encryption/decryption, such as hashing, AES, and signature verification have been verified standalone in RTL simulations. The full secure boot application has then been implemented on the FPGA, where performance numbers can be extracted.

Feature	**Component ID**	**Component name**	**Layer**
Security	WP3T32-10	TL2AXI adapter	Node
	UC3-01	Secure boot process	Node
	UC3-02	Encryption, decryption	Node / application
Connectivity	UC3-03	Wireless connectivity	Node
	WP3T36-03	Nuttx on RISC-V	Service-orchestration
Low Power	WP3T32-02	PULPissimo platform	Node
	WP3T32-08	Real-time cache	Node
	WP3T32-11	Smart Interrupt unit	Node
	WP4T41-03	Low power services	Node / application
	WP3T32-05	ML inference demo	Application
	WP3T32-04	FreeRTOS	Node

Table 10: Component Id, component name, abstraction layer used to implement the FRACTAL features SECURTIY, CONNECTIVITY and LOW POWER.

6.3 Demonstrator setup

Two demonstrator setups are presented in the following. The first is used to showcase the security components, and the second for the low power and wireless connectivity components.

6.3.1 Demonstrator setup for security components

The security components are implemented on the genesys2 FPGA board. More information about the demonstrator setup is presented in Section 6.2.1 of deliverable D7.2.

On the FPGA, the execution times of the secure boot process can be precisely profiled using the cycle counter of the processor. The execution time of the secure boot process depends mainly on the time required to create a hash, and the time to verify the signature. Two signature verification techniques have been analyzed. The first is a classical 3072-bit RSA signature verification, and the second an Elliptic Curve Digital Signature Algorithm (ECDSA) with a 256-byte signature. Both algorithms offer similar security levels.

To speed up the signature verification, the Elliptic Curve Cryptographic (ECC) functions have been implemented on the OpenTitan Big Number accelerator (OTBN), where a signature can be verified within 4.7ms. The execution times of the different tasks are given in Table 1.

Task	# Cycles	ms @ 100MHz
Hashing 10.4 KiB ROM_EXT partition	117406	1.17
Hashing 4 MiB application image	45616207	456.16
Verify 3072 bit RSA signature with CPU	16472957	164.73
Verify 3072 bit RSA signature with OTBN	1007579	10.08
Verify 64 byte ECDSA-256 signature with OTBN	476576	4.77

Table 11: Profiling results of key functions.

During the first stage of the secure boot process a hash of the ROM_EXT partition is computed in 1.17ms, followed by a 10.08ms RSA signature verification. In the second stage the application image of 4MiB is hashed in 456.16ms, followed by ECDSA verification of 4.77ms. The full secure boot process with a 10kiB bootloader and a 4MiB application image can be completed in 519ms and is well below the 1s target even if an intermediate third stage would be added to the boot process.

6.3.2 Demonstrator setup for wireless connectivity, low power components and ML inference

The demonstrator setup of section 6.2.2 in deliverable D7.2 has been updated. The fractal board has been interfaced with a camera as shown in Figure 1.

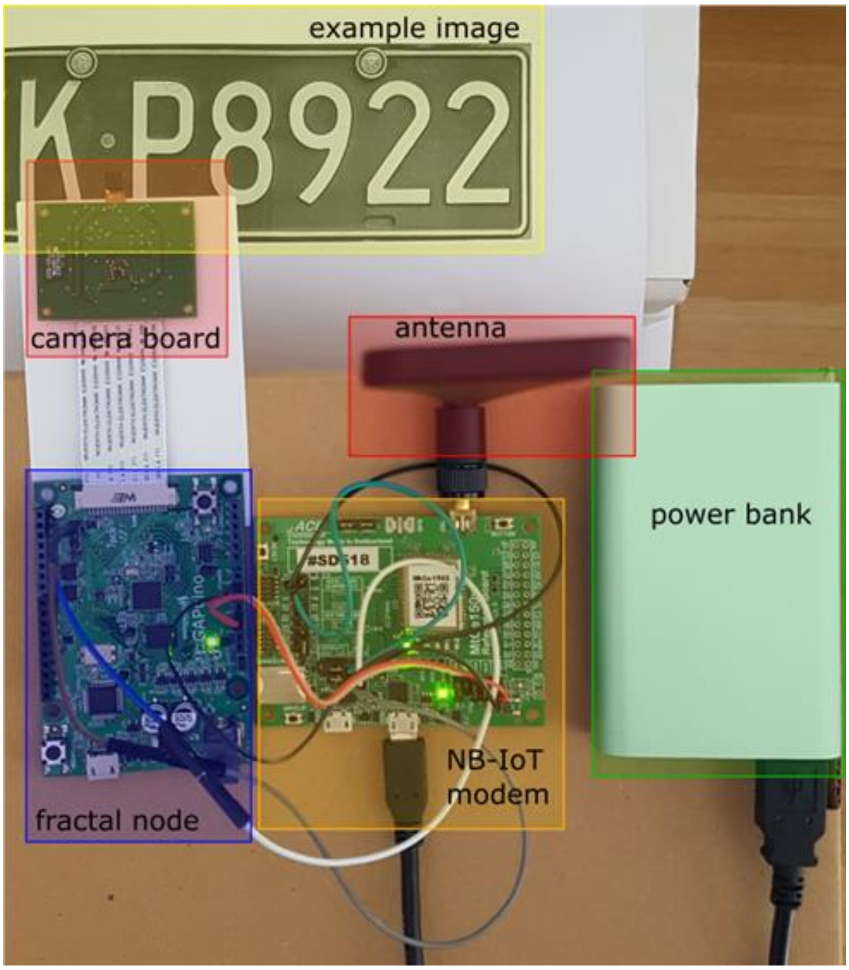

Figure 25: Demonstration setup with camera, pulpissimo type fractal node and the NB-IoT modem.

A ML inference demonstration has been implemented in software and executed on the fractal node. The application goes through the following steps:

1. Take a picture upon request (GPIO)
    a. Store image in flash
2. Launch a two-step inference algorithm:
    a. Detection of plate
    b. Recognition of letters and numbers
3. Establish a NB-IoT connection
4. Transmit extracted information to a server where it can be displayed
5. Go to sleep

The neural network is trained for license plate recognition. In a first step it detects the plate, and in a second step it recognizes the letters and numbers within the plate. This implementation has been chosen because of an existing data set and because the requirements are very similar to the requirements of the use case.

Different versions of the application have been profiled. The baseline implementation uses a retention sleep mode where the full memory content was retained. In version A) an optimized deep sleep mode is used which stores relevant data on the flash upon entering deep sleep and restores the data upon wakeup. Version B) uses an optimized version of this deep sleep mode. Version C) is very similar to version B) but activates a circuit that automatically controls a top-level clock gate that allows to save power by gating the clock in short idle periods. All versions are implemented on the two-board setup as shown in figure.

The unoptimized deep sleep power can be measured on the board and is 2.5uA at battery voltage. The current consumption of the optimized deep sleep mode is determined with a leakage power analysis in the digital implementation tool. According to this analysis the deep sleep power is expected to go down to 1.7uA.

The execution time and active power consumption of the first two steps is constant, the remaining steps on the other hand depend on different factors such as the signal to noise ratio (SNR). In the subsequent measurements the average power and execution time was averaged over 10 runs. The active time is then multiplied by the number of readings per day. The average execution time of the loop was 46.6 seconds and the average current consumption of the whole system in active state was measured at 40.9mA when automatic top-level clock gating was turned on, and 43.1mA when it was turned off. The impact of clock gating can be demonstrated on the modem which has a top-level clock gate that allows to automatically gate the clock when all its stakeholders are idle. It has been measured that the power can be lowered by 14.1% in idle state.

Table 2 shows the execution times, average active current consumption, and deep sleep current consumption. With a battery capacity of 2200mAh, version C) can achieve a battery lifetime of 5+ years. The lifetime can be further increased by limiting the readings to once per day as shown for version D)

Version	**No. of readings / day**	**Active time/day [s]**	**Average active current [mA]**	**Sleep current [uA]**	**Lifetime [years]**
Baseline	2	99.2	43.1	700	0.33
A) + deep sleep	2	99.2	43.1	2.5	4.82
B) + optimized deep sleep	2	99.2	43.1	1.7	4.90
C) + top level clock gating	2	99.2	40.9	1.7	5.16
D) only 1 reading	1	49.6	40.9	1.7	9.9

Table 12: Battery lifetime computation of different versions.

Finally, merging the two platforms into one would allow to bring the current consumption further down and will be explored in the future.

6.4 Validation & assessment of KPIs

The KPIs with respect to SECURITY have been fulfilled by using the open-source hardware IPs of OpenTitan and implementing a secure boot process as shown in section 6.3.1.

The deep sleep power consumption of only 1.7 uA@VBAT exceeds the 10uA requirement and competes with other commercially available MCUs. Top-level clock gating alone allows to reduce the active current by 14.1%.

FRACTAL feature Security	
UC3 contributes to the implementation of a secure boot process with the goal of making the FRACTAL node secure and to be able to encrypt user data directly on the edge in an energy-efficient way.	

Technical KPIs	• Possibility to verify a firmware during the boot process (**firmware verified during boot**).
	   ○ Result: Functionality supported
	• The verification of a 4 MB firmware should take **less than 1s**.
	   ○ Result: 519ms
	• Being able to encrypt/decrypt data on the edge device (**en/decryption capability**)
	   ○ Result: Functionality supported
	• AES256 encryption/decryption of a **16 kB block in < 100ms**
	   ○ Result: Open
	• AES256 encryption/decryption of a 4 MB firmware **in less than 1 minute**.
	   ○ Result: Open

Business KPIs	• Having a secure boot process is a big selling point for an SoC ○ Result: Functionality supported

Table 13: KPIs to assess the FRACTAL feature SECURITY.

FRACTAL feature Low Power	
As part of UC3, the power consumption of the FRACTAL node will be optimized such that the low-end node based on Pulpissimo can achieve a multiyear battery lifetime. The key to a long battery lifetime is an ultra-low power deep sleep state.	

Technical KPIs	• Deep sleep current consumption **< 10 μA**
	   ○ Result: 1.7uA
	• Idle power reduction of **> 10 %** thanks to clock gating
	   ○ Result: 14.1% with top-level clock gating
	• Battery lifetime with a 2200mAh in the order of **> 5 years**
	   ○ Result: 5.16 years with 2 readings / day

Business KPIs	• Reducing the area of an IC not only reduces its power consumption, but also its price (**>20% area reduction**) ○ Result: Analysis open

Table 14: KPIs to assess FRACTAL feature LOW POWER.

Use case 3 supports the open-source projects and builds on a set up open-source hardware and software modules as shown in the table below:

UC3: SmartMeter	**License**
Open-Source Software / Libraries	
FreeRTOS **(https://github.com/pulp-platform/pulp-freertos)**	Apache v2.0 (permissive)
OpenTitan Software (https://github.com/lowRISC/opentitan)	Apache v2.0 (permissive)
Open-Source Hardware	
OpenTitan (https://github.com/lowRISC/opentitan)	Apache v2.0 (permissive)
PULPissimo (https://github.com/pulp-platform/pulpissimo)	Solderpad v0.51 (permissive)

Table 15: Open-source products alongside the used licences for UC 3.

6.5 Path to Exploitation

ACP's modem already offers wireless connectivity, but it lacks processing power for edge computing. Thanks to the inclusion of the fractal node, it can now do simple image processing tasks such as feature extraction of an image as shown in the demonstrator of section 6.3. This makes it possible to extract the relevant data directly on the edge, and then only transmit the relevant data over the wireless network rather than raw data such as images. This opens the field for a set of future low-power applications which rely on battery operation.

Security features (secure boot process) will be exploited in future versions of our modem to make it secure. The secure boot process allows to have an on-chip root of trust which is an important feature for many applications.

Low power features (additional clock + power gating) have already been incorporated into the latest revision of the modem to reduce its power consumption during active and idle states. This will extend the battery life of all applications.

7 UC4 – Low Latency Object Detection

7.1 Addressed features

The object detection accuracy that can be achieved with the use of AI has shown to be a very promising solution. Such a technology can be very beneficial on a variety of implementations of the production line in industry, where the produced products need to be observed for possible faults or to detect their location on the production line. The use case addresses the capability of Fractal platform for recognizing the surrounding and with that demonstrating the COGNITIVE AWARENESS feature of the platform.

7.1.1 FRACTAL feature Cognitive Awareness

Computer vision is a crucial component for cognition to extract meaningful information from the surrounding. Recognizing the changes in the environment gives the Fractal platform capability to react to the state changes that are happening around the system. This is achieved through the implementation of the AI algorithm running on Fractal edge as inference and trained in Fractal cloud when new object needs to be added. Thus, the use case observes the surrounding and triggers immediate actions related to the occurred change.

7.1.2 FRACTAL feature Cloud Communication / Connectivity and openness

The use case partially also verifies the CONNECTIVITY and openness as features. The CONNECTIVITY is demonstrated through data exchange between Fractal node and Fractal cloud when training of the inference is needed while the openness is demonstrated with the use of open-source RISC-V processor implementation and Darknet framework with YOLO algorithm for object recognition.

7.2 Building block integration and verification

The following table shows the list of components needed to build the platform for demonstration of the use case.

feature	building block / component	Component name	layer
Cognitive awareness	WP3T32-01	HW accelerator	node layer
	WP3T33-01	Ariane RISC-V for ZCU102	node layer
	WP3T35-01	SW-driver for HW accelerator	application
	WP3T32-02b	Ariane for Linux capable RISC-V platform	node layer
	WP3T36-01	Linux for CVA6	application
	WP5T52-04-05	Datasets version control	application
	WP5T52-04-06	Feature storage	application
	WP6T61-12	Neural network toolkit	application

Table 16: Components used to implement the FRACTAL feature COGNITIVE AWARENESS.

Three of the components are used to build the node layer and two to build the application for the edge part of the platform, while the last three are software components composing the application of the Fractal cloud for this use case. Below we describe the role of each component within the use case.

- WP3T32-01 is **HW accelerator** for execution of the convolutional layer. The component is connected to the AXI bus and behave as a slave bus node. Its purpose is to perform dot operation on data provided locally through memory DMA.
- WP3T33-01 is **CVA6 RISC-V for ZCU102**. The component is ported from Xilinx Kintex-7 FPGA board to ZCU102 SoC FPGA board due to the size of the programable area available in the latest board and for the possibility to have and compare the CVA6 and ARM cores on the same platform.

- WP3T35-01 is **SW driver for HW accelerator**. It is developed to manage and monitor the work of the HW accelerator. The driver organizes the data transfer between main memory and local memory on hardware accelerator, triggers the accelerator to perform dot operation and monitors the execution.
- WP3T32-02b is **Ariane for Linux capable RISC-V platform**. The component is the RISC-V processor available from OpenHW group.
- WP3T36-01 is **Linux for CVA6**. The OS is a generated with Buildroot and can run on 32- and 64-bits platforms. It is a simple Linux OS with only the most necessary components required for running the AI inference on Fractal edge node.
- WP5T52-04-05 is **Dataset version control**. The component is a storage for deferent dataset used to perform training of the AI algorithm on Fractal cloud.
- WP5T52-04-06 is **Feature storage**. It keeps the previous version of AI algorithm trained in the Fractal cloud.
- WP6T61-12 is **Neural network toolkit**. It is Darknet framework, an open-source tool used for performing training of the Tiny-YOLO algorithm on Fractal cloud and running its inference on Fractal edge node.

7.3 Demonstrator setup

The use case demonstration is performed on Fractal edge node and Fractal cloud. The inference is executed on edge while the training on the cloud. The edge is setup on FPGA board consisted of RISC-V CPU HW accelerator, DMA and system memory (Figure 26). The Cloud is triggered only when it is needed to train the AI algorithm for new objects to be detected. The details on implementation are given in the previous deliverable D7.2.

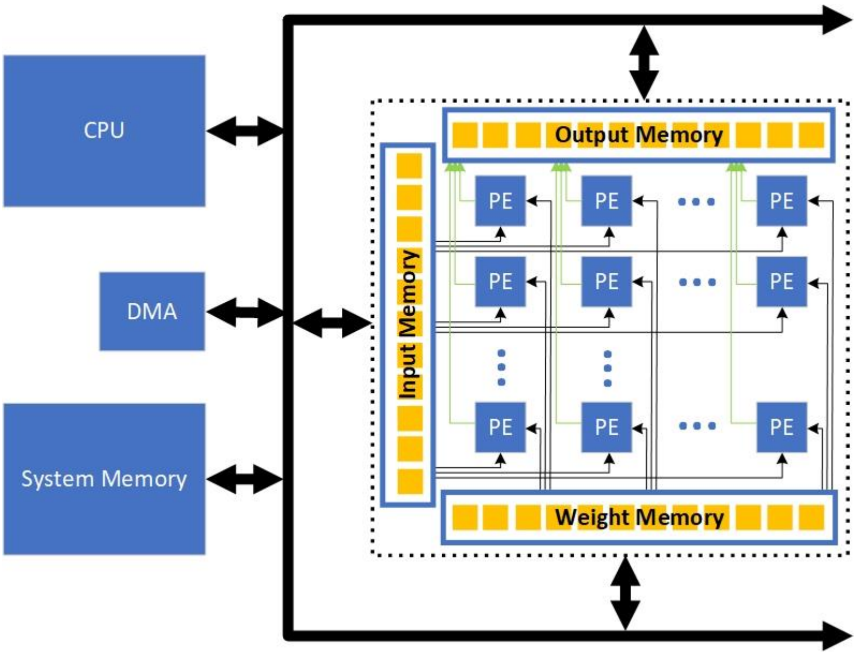

Figure 26: Hardware architecture of Fractal edge.

For demonstration of the use case, we use a set of PCB board images to evaluate the training of AI algorithm for detection of electronic components on a PCB board. Once this is confirmed we use a set of inputs to evaluate the timing required to perform convolution operation on RISC-V processor. On the next step, the same process is repeated with the only difference that now the convolution operation is performed on HW accelerator. The results from both outcomes are compared to estimate the benefits gained from the presence of the HW accelerator when convolution operation is performed.

7.4 Validation & assessment of KPIs

With detection of the object on the running video stream, the use case validates the cognitive awareness feature provided by FRACTAL platform. The assessment of the achieved speedup is one of the main parameters to be evaluated. In the following we provide the set of equation used for determining the main component to be evaluated and used as a parameter for assessment of the use case KPI.

Equation (1) present the total execution time (t_{exe}) consisted of preprocessing time (t_{pre}) needed to prepare the image in an adequate format, time to perform convolution (t_{con}) and time for post processing (t_{pos}) of the image by marking and labeling the recognized objects on the images.

$$t_{exe} = t_{pre} + t_{con} + t_{pos} \tag{1}$$

The time required for convolution (t_{con}) is the main component of our analysis. When the convolution is performed on a singe CPU it can be represented as a sum all simple dot operation (2).

$$t_{con\_cpu} = \sum t_{op} \tag{2}$$

On the other hand, the convolution time preformed in an accelerator is described on equation (3). Apart from time for dot operations, the evaluation takes in addition the time required to transfer the input data (t_i) from system memory to the local one, time to transfer weights (t_w) and time to transfer the output (t_o) back from local accelerator memory to the system memory.

$$t_{con\_acc} = t_i + t_w + \sum t_{op} /(n*p) + t_o \tag{3}$$

The assessment of the KPI is based on the equation (4). The goal is to evaluate the achieved at least twice speed-up on convolution operation performed between CPU and accelerator.

$$(t_{con\_cpu} / t_{con\_acc}) > 2 \tag{4}$$

7.5 Path to exploitation

The first outcome of the use case is the technology for hardware accelerator and its integration as part of the SoC. This technology will be integrated into HLS toolchain of Siemens.

The second outcome is the software developed for control of the data flow between CPU and accelerated and within accelerator as well. This component will also be integrated into HLS toolchain of Siemens.

The third outcome is the platform with RISC-V processor that will serve as a starting point for future RISC-V designs of Siemens portfolio.

The outcome from the use case will be considered to be part of an industrial production line where visual defects on products can be detected with this technology or in the production line with industrial robot arm where the technology will be used to drive the arm to the detect object.

8 List of figures

9 List of tables

10 List of Abbreviations

Abbreviation	Term
A3C	Advantage Actor-Centric
ACAP	Adaptable Compute Acceleration Platform
AUTOSAR	Automotive Open Systems Architecture
BEV	Battery Electric Vehicle
CNN	Convolutional Neural Network
CPU	Central Processing Unit
ECC	Elliptic Curve Cryptographic
ECDSA	Elliptic Curve Digital Signature Algorithm
FMU	Functional Mockup Unit
FPGA	Field Programmable Gate Array
FPN	Multi-scale models
GPIO	General Purpose Input Output
HW	Hardware
IoT	Internet of Things
IOU	Input / Output Unit
ONNX	Open neural network exchange
OTBN	OpenTitan Big Number accelerator
MPC	Model Predictive Control
PPO	Proximal Policy Optimization
RISC	Reduced Instruction Set Computer
RL	reinforcement learning
SaaS	Sofware as a Service
SNR	Signal to Noise Ratio
SoC	State of Charge
SW	Software
TRL	Technology Readiness Level
UAV	Unmanned Areal Vehicle
UC	Use case
WP	Work Package

11 Appendix A: FRACTAL components

Component ID	Component / Subcomponent name
WP3	
WP3-AI	**AI accelerator (hardware and software support)**
WP3T32-01	HW accelerator (SIEFRACC)
WP3T32-05	ML inference demo PULPissimo
WP3T32-07	Age and Gender identifier at the edge
WP3T32-10	VERSAL accelerator building-blocks
WP3T33-01	Ariane RISC-V for ZCU102
WP3T34-03	VERSAL Model deployment layer
WP3T35-01	SW driver for HW accelerator
WP3T35-02	Accelerator Adaptation to AI library
WP3T35-03	LEDEL (Low Energy EDDL)
WP3T35-04	Deep learning based automatic iris diagnosis
WP3T35-05	Idiom Recognition
WP3-CPU/OS	**CPU and OS support**
WP3T32-02	PULPissimo platform for IoT applications
WP3T32-08	Real-time aware caches
WP3T32-11	Smart Interrupt distribution system
WP3T32-12	Security services - TL2AXI adapter
WP3T32-02b	Ariane for Linux capable RISC-V platform
WP3T32-03	PULP trainings
WP3T32-04	FreeRTOS port to PULP
WP3T33-03	CVA6 (former Ariane) RISC-V core
WP3T36-01	Linux for CVA6 (former Ariane)
WP3T36-02	Load Balancing Module
WP3T36-03	Nuttx on PULP
WP3-Safety	**Safety and security features for CPU**
WP3T31-01	Edge-oriented monitoring unit
WP3T31-02	Interconnect to support Accelerators integration
WP3T31-03	Safety and security hardware support
WP3T32-06	Redundant Acceleration Scheme
WP3T32-09	Runtime Bandwidth Regulator
WP3T34-01	Driver for the edge-oriented monitoring unit
WP3T34-02	Drivers for the SW diverse redundancy library

WP4

WP4T41	**Low Power Services**
WP4T41-01	Data Compression for Low-Power Services
WP4T41-02	HATMA
WP4T41-03	Low Power services for PULP systems
WP4T41-04	VERSAL RPU access for Power Services
WP4T41-05	Agreement protocol for Low-Power Services
WP4T41-06	VERSAL Isolation Design - Functional Safety
WP4T42	**AI-Based Scheduling**
WP4T42-01	Capabilities for AI supported adaptability in PULP
WP4T42-02	VERSAL RPU access to AI acceleration
WP4T42-03	Scenario Generator
WP4T42-04	GA-Scheduler
WP4T42-05	AI-Scheduler Model
WP4T42-06	Schedule Verifier
WP4T42-07	Hierarchical Metascheduler
WP4T43	**Safety Services**
WP4T43-01	Performance monitoring services
WP4T43-02	Safety services for PULP systems
WP4T43-03	SW diverse redundancy library
WP4T43-04	ATTNoC
WP4T43-05	Redundant Acceleration Scheme Safety Analysis
WP4T43-06	FPGA Fault-injector
WP4T43-07	Safety Case
WP4T43-08	Seamless redundancy for ATTNoC
WP4T43-09	Safety functions consideration during ML
WP4T43-10	Safety functions (Reinforcement learning)
WP4T43-11	Time-Triggered Extension Layer for VERSAL NoC
WP4T43-12	Safety Analysis
WP4T43-13	Safety Analysis
WP4T44	**Security Services**
WP4T44-01	Security services for PULP systems
WP4T44-02	OS Security Layer
WP4T44-03	Security services at application and network layers
WP4T44-04	Security Assesment
WP4T44-05	IoT Gateway
WP4T44-06	GDPR Compliance
WP4T44-07	Node monitoring and system status
WP4T44-08	TLS Implementation on containers
WP4T44-09	Runtime security
WP4T44-10	LEDEL validation

WP5

WP5T52-01	**Cloud Platform Data Ingestion**
WP5T52-01-01	Data Ingestion Service
WP5T52-02	**Cloud Platform Raw Data Storage**
WP5T52-02-01	Raw data Object storage service
WP5T52-03	**Cloud Platform Data Transformation**
WP5T52-03-01	Data transformation
WP5T52-04	**Cloud Platform Repositories**
WP5T52-04-01	Models version control
WP5T52-04-03	S3 compatible data storage
WP5T52-04-05	Datasets version control
WP5T52-04-06	Feature storage
WP5T52-04-07	Images repository
WP5T52-04-08	Model repository
WP5T52-04-09	Machine learning pipeline
WP5T52-04-10	MLBuffet as a Cloud Model Storage
WP5T52-05	**Cloud Platform Orchestration**
WP5T52-05-01	MLBuffet as a Cloud Model Orchestrator
WP5T52-05-02	Data pipelines and workflows orchestrator
WP5T52-06	**Cloud Platform Models Serving**
WP5T52-06-01	Model preparation for Fractal Edge (VERSAL Xilinx Vitis AI)
WP5T52-06-02	ML pipeline connection to model repository
WP5T52-07	**Cloud Platform Infrastructure**
WP5T52-07-01	Kubernetes-based cloud platform container orchestrator
WP5T52-07-02	Cloud container orchestrator services access
WP5T54-01	**AI on the Edge**
WP5T54-01-01	MLBuffet
WP5T54-01-02	Training module for MLBuffet
WP5T54-02	**Orchestrators**
WP5T54-02-01	Docker Swarm
WP5T54-02-02	Kubernetes-based Container Orchestrators for the Edge
WP5T54-03	MLOps Toolchain
WP5T56_01	People detector example

WP6

WP6T61-01	Edge ML API
WP6T61-02	Edge API
WP6T61-03	Data Analysis
WP6T61-04	Data Pre-processing
WP6T61-12	Neural Network Toolkit
WP6T62-01	Data Ingestion
WP6T62-02	Federated Data Collection
WP6T62-03	Run time Manager
WP6T61-05	ML/AI Tools
WP6T62-06	Orchestration