Digital environments – Final Version (full ecosystem: environment & benchmarks)

Deliverable D1.4

Distribution level: Public

Version: 1.0

Date of delivery: 30/09/2026

AI4REALNET has received funding from European Union's Horizon Europe Research and Innovation programme under the Grant Agreement No 101119527, and from the Swiss State Secretariat for Education, Research and Innovation.

List of authors

Authors	Institution
Milad Leyli-abadi	IRTSX
Maroua Meddeb	IRTSX
Abderrahman Ait-Said	IRTSX
Giulia Leto	TUD
Clark Borst	TUD
Jack Agterdenbos	TUD
Javier Crespo	TUD
Ricardo Bessa	INESC TEC
Olga Klyagina	INESC TEC
Duarte Dias	INESC TEC
Federico Calà	INESC TEC
Bruno Lemetayer	RTE
Antoine Marot	RTE
Manuel Meyer	Flatland

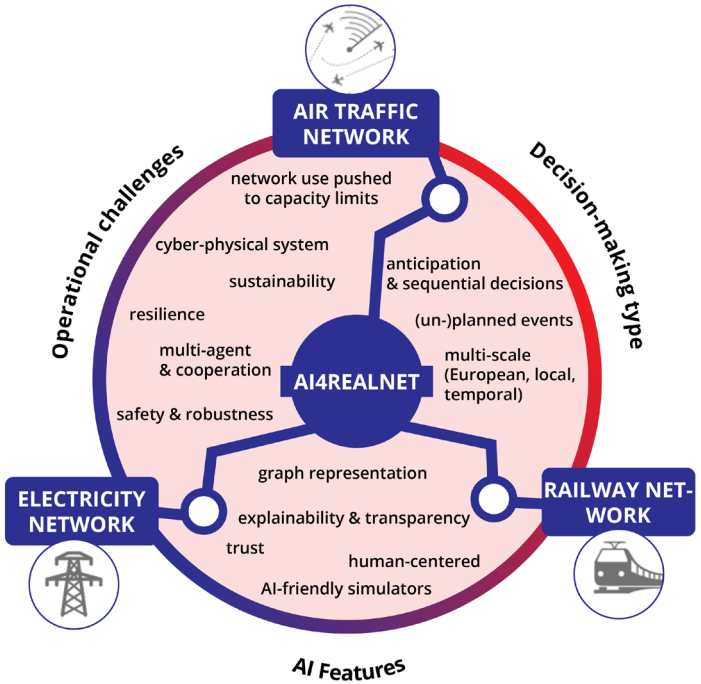
List of authors

Authors	Institution
Anatolii Korol	Ukassel
Kurt Brendlinger	Fraunhofer
Mohamed Hassouna	Ukassel
Manuel Renold	FHNW
Janick Marti	FHNW
Julia Stadelmann	FHNW
Anton Fuxjäger	EnliteAI
Edi Muskardin	EnliteAI

Background: AI4REALNET

GOALS

- Develop the next generation of decision-making methods powered by supervised and reinforcement learning, which aim at trustworthiness in AI-assisted human control, human-AI co-learning, and autonomous AI
- Boost the development and validation of novel AI algorithms via 3 existing open-source AI-friendly digital environments



List of software modules, data & resources

- Grid2op:
  - AI4REALNET/grid2op: platform to model sequential decision making in power systems
  - AI4REALNET/pypowsybl2grid: integration between Grid2op and PyPowSyBl backend (Python library for power grid modelling and simulation)
- Flatland:
  - AI4REALNET/flatland-rl: Simulation environment for modeling railway scheduling and rescheduling problems using reinforcement learning and operation s research. Support for Gymnasium[1], RLlib[2], and PettingZoo[3] Environments. Generation of synthetic railway environments.
  - AI4REALNET/flatland-baselines: Baseline models for flatland.
  - AI4REALNET/flatland-scenarios: Scenarios for the railway use cases.
  - AI4REALNET/flatland-book: Conceptual and technical documentation of the Flatland environment.
- BlueSky:
  - AI4REALNET/BlueSky-Gym: A Gymnasium[1] style library for standardized Reinforcement Learning research in Air Traffic Management.
  - Use cases in BlueSky scenarios (proprietary data stored in the AI4REALNET GitLab, available upon request)
  - AI4REALNET/ATMSectorization: an interface for transparent decision making for sectorization, supporting use case 1.
  - AI4REALNET/bluesky, readme: plugins for testing BlueSky-Gym RL models on realistic scenarios
  - FAB ATM test runner: sample test runner that connects the testing of RL in BlueSky with the FAB client
- InteractiveAI:
  - AI4REALNET/InteractiveAI : Generic platform for human interaction with complex networks
  - Specific HMI modules for each industrial domain (ATM, Power grid, Railway)
  - Interfaces with each digital environment (BlueSky, Grid2Op, Flatland)
  - https://github.com/AI4REALNET/InteractiveAI-API Generic interoperability APIs to ensure communication with different digital environment
  - Instantiation of the generic interoperability APIs with the Railway Use Case using Maze Flatland AI agent

[1] https://gymnasium.farama.org/
[2] https://docs.ray.io/en/latest/rllib/index.html
[3] https://pettingzoo.farama.org/index.html

Grid2Op: Power Grid Digital Environment

Developer: RTE

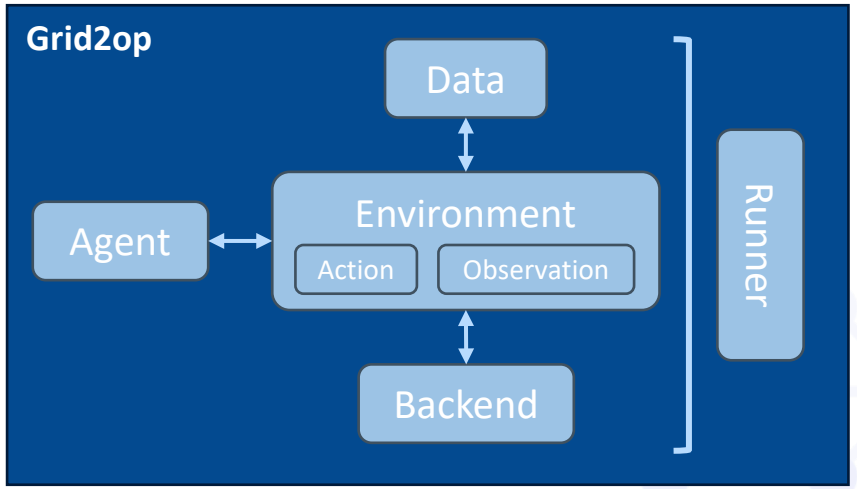
General description

Develop / train / evaluate performances of an agent acting on a Power grid

- Simulates model sequential decision making made by human operators to ease the research on sequential decision making applied to power systems
- Fully compatible with standard Gymnasium Reinforcement Learning API
- Abstracts the computation of the cascading failures
- Allows running same code with multiple power flows simulators (PandaPower, PyPowsybl, LightSim2Grid)
- Execute one agent on multiple independent scenarios in parallel
- Can be fully customized to serve different purposes (and not only Reinforcement Learning)
- Developed in Python
- Full documentation available at https://grid2op.readthedocs.io/en/latest/

Block diagram

- **Environment** is a representation of the power grid with which an Agent interacts (Generators, Loads, Powerlines, Shunts, Storage units, Substations)
- **Agents** can perform **Actions** on the environment, and receive **Observations** that correspond to its current or forecasted state
- Environment state evolves using **Data** (Time series) corresponding to injections (generators, loads) and structural information (planned outage, hazards)
- **Backend** allows calculating the physical state of the environment (Power grid, flow, intensity, etc.)
- The evaluation of agents can be performed by **Runner**, allowing corresponding data to be logged



Other experimental modules exists as part of the "GridAlive" ecosystem, but are not used in the project:

- Grid2Game (basic human GUI)
- Grid2Viz (visualization GUI)
- Grid2Bench (KPIs and benchmark)

Input data

- Environment
  - environment time series can be read from CSV (possibly by chunks to improve IO runtime)
  - “grid.json” is the full description of the powergrid characteristics, to be used by the backend
  - “grid_layout.json” is the geographic description of the powergrid for display
  - “config.py” is the configuration file
- Agent
  - Abstract class grid2op.Action.BaseAgent API can be implemented/extended to create custom agents
  - To perform their actions, Agents
    - receive grid2op.Reward.BaseReward and grid2op.Observation.BaseObservation data from the grid2op.Environment API
    - interacts with the grid2op.Action.ActionSpace API from the grid2op.Environment API
  - Abstract class grid2op.Action.BaseAction API can be implemented/extended to create custom actions
- Backends : Since release v1, PyPowsybl backend is available thanks to pypowsybl2grid package

Output data

- Environment
  - State of the environment is described by the grid2op.Observation.BaseObservation object
  - Observations are described by more than 30 attributes available in Observation API
  - Visualized using basic plotting features

    Grid plot image can be generated using backend functionalities

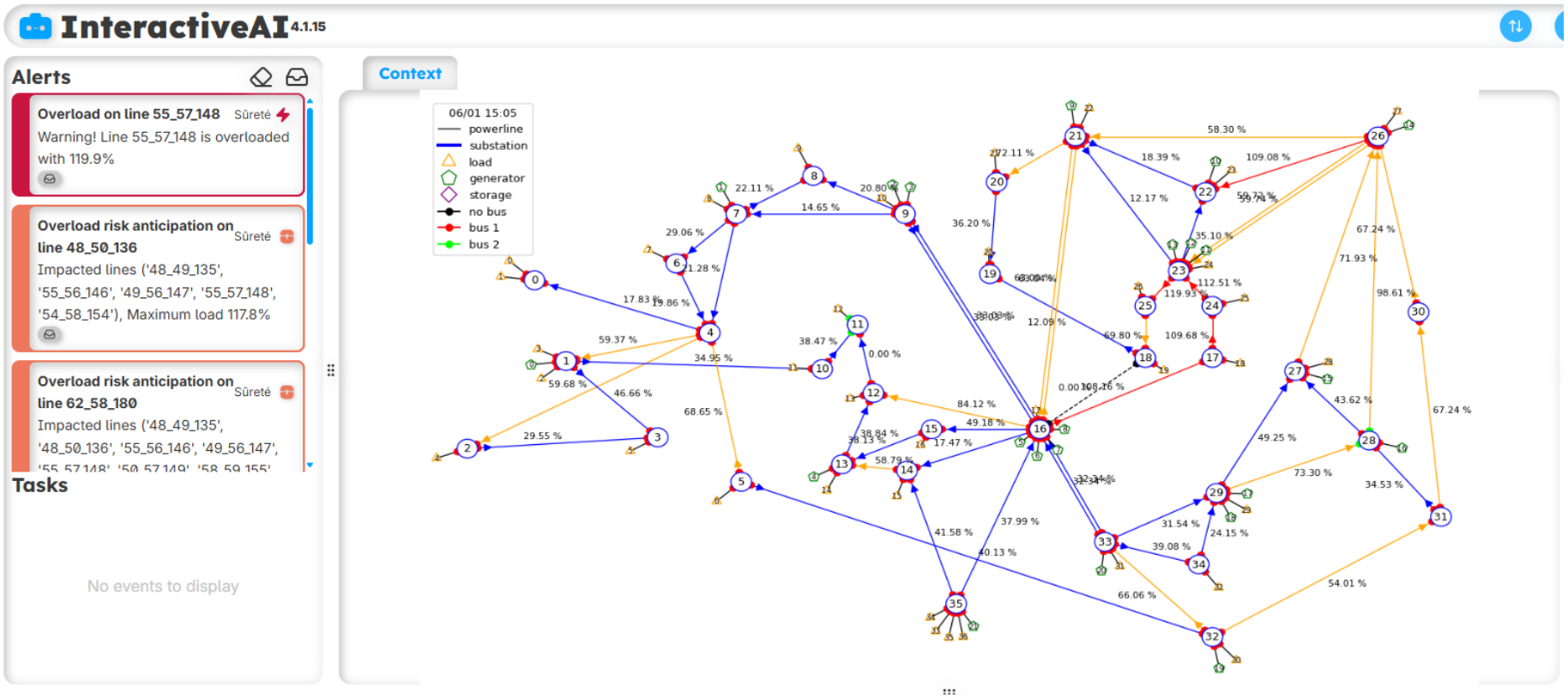
    Grid2Op allows for more advanced display features to be developed, for example in an advanced HMI framework
  - Exported in standard file formats (for example, iidm(*) thanks to the PyPowsybl backend)
- Runner : results of an evaluation are saved in standard JSON and NumPy (.npz) formats
- EnvRecorder added in v2:
  - Environment recorder for capturing and storing environment data
  - Observations saved into Parquet files for later analysis

(*) internal grid model initially developed under the iTesla research project that was funded by the European Union 7th Framework Programme (FP7)

New Features (final release)

- New method to generate data on the “small” (36 nodes) environment

Short demonstration (photo/video)



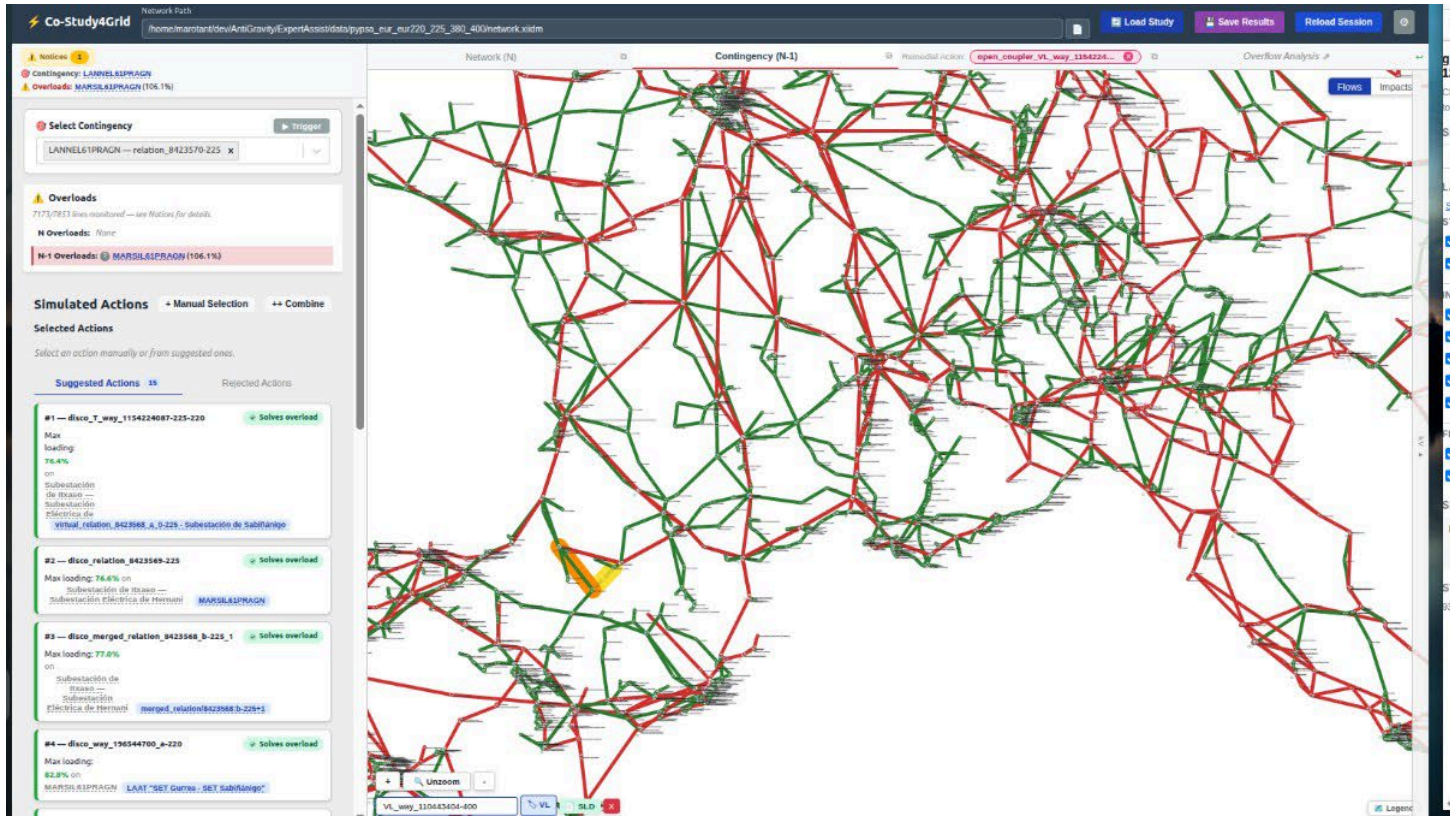
Co-Study4Grid

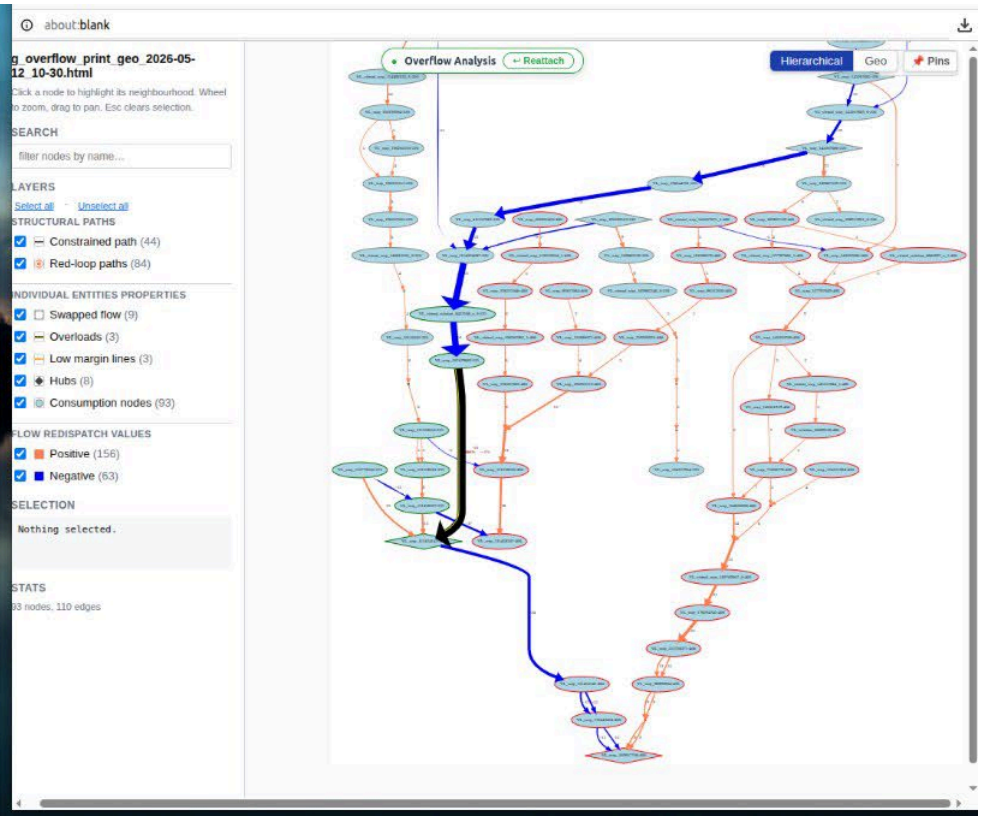
A complementary web application for contingency study and remedial action search that implements the supportive framework mindset from AI4RealNet

A showcase of fast prototyping allowed by vibe coding, that could be further coupled or integrated with InteractiveAI

- Contingency study & remediation
  - Familiarity: Operators can simulate element disconnections, visualize and analyze network overflows, make their own hypothesis and simulate them similarly to their study tool today
  - Recommendations: request prioritized and diverse remedial actions after model selection (default: expert_op4grid_recommender library)
  - Reflective Interactions & Exploration: wide range of interactions to inspect actions, relations, influences on network or tabular representation, or even combine actions
  - Replay: Reload existing study, with related recommendations and preferred choices
- Visualization
  - Four synchronized tabs — Network N, Contingency N-1, Remedial Action, Overflow Analysis with Flow vs Impact visual modes
  - Network-Area Diagrams (NAD) with interactive overflow analysis, highlighted remedial actions
  - Single-Line Diagrams (SLD) with highlight of impacted switches and coupling breakers

Co-Study4Grid





Flatland: Railway Digital Environment

Developer: Flatland Association

General description

The Flatland environment provides simplistic representation of a rail network on a grid world to address the vehicle rescheduling problem (VRSP)

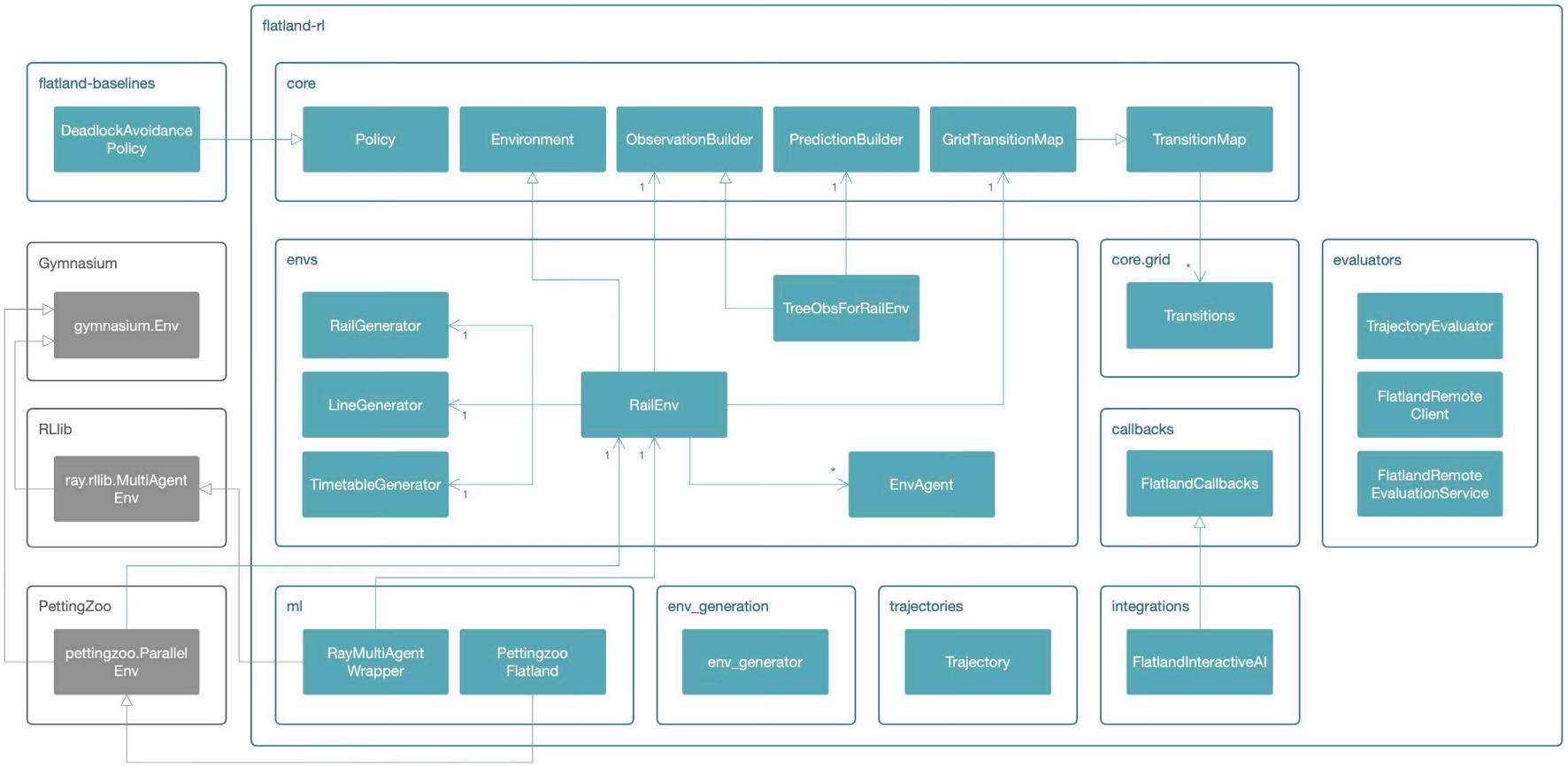
- Flatland is used to develop reinforcement learning (RL) solutions to the VRSP
- Trains are agents with a limited action space (↑ , ← , → , ✕ , ⟳)
- Agents have schedules for their origin, destination and intermediate stops
- Railway network includes switches, slips, crossings and over-/underpasses
- Translation from grid representation of the network to a graph representation is implemented
- Agents can have speed profiles, reflecting different train classes (passenger, freight, etc.)
- Agents can be disrupted (in malfunction)

flatland-rl flatland-scenarios flatland-baselines flatland-book

General description

- Starter kit for easy access and as simple benchmark is provided
- Standard observations and rewards provided
- Trajectories are stored for benchmarking and regression testing
- Real-world example scenarios and generated railway environments are provided
- Integration with InteractiveAI API (user interface) provided
- Multi Objectives capability available
- Compatible with standard Gymnasium Reinforcement Learning API
- Compatible with RLlib and PettingZoo API
- Developed in Python
- Full documentation: Flatland book

Block diagram



flatland-rl is the main package of Flatland and includes:

- **core** (base classes and functionality for simulations and reinforcement learning),
- **envs** (railway environment specific functionality),
- **callbacks** and **integrations** (for extended interactions, e.g., InteractiveAI),
- **ml** (Gymnasium, RLlib and PettingZoo support),
- **env_generation** and **trajectories** (environment generation and simulation run persistence)
- **evaluators** (functionality for scenario evaluation)

flatland-baselines provides a heuristic based policy baseline

New Features (final release)



- Implemented new reward and reward component (used for railway competition)
- Data processing for new visualization element (Marey diagram) used in HMI
- Improved rail generator
- Extended persistence (trajectories)
- Overall cleanup and performance improvements

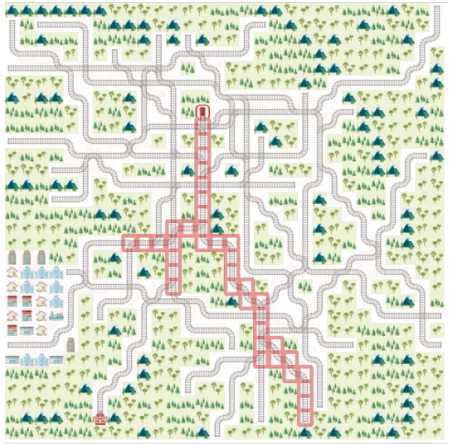
Input data



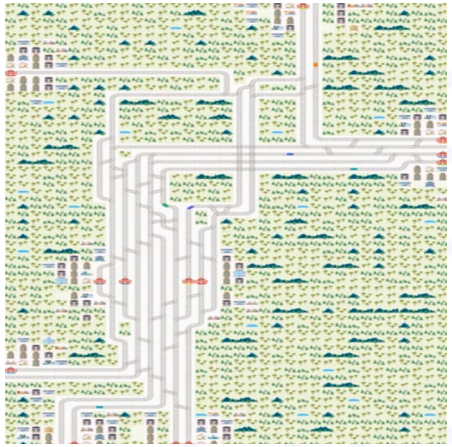
- Environment
  - Environments, i.e., railway network, train stations, schedules, lines, etc., can be specified manually or being generated based on parameters (grid size, number of agents, network characteristics, number of stations, etc.) in Python scripts.
  - Environments can be loaded from PKL (Pickled Python Objects) files.
  - Simulation runs, i.e., a sequence of environment states and actions (trajectories), can be loaded from PKL and TSV (tab-separated values) files.
- Agent
  - Agents interact with the environment by providing one of 5 actions (move forward, move left, move right, stop, do nothing) at each time step. With dynamic speed: acceleration by move forward action, deceleration by stop action.
  - Agents have access to an observation and rewards at each time step.
  - Models for the Agents can be loaded from PKL files.
  - Gymnasium, RLlib, PettingZoo APIs are available for agent-environment interaction.

Output data

- Environment
  - Environments can be saved as PKL files.
  - State, observations and rewards are provided for each time step.
  - Trajectories can be saved as PKL and TSV files.
  - Renderings (videos / images) of simulation runs including observations and other information can be generated and saved in standard media file formats.
  - Events compatible with InteractiveAI framework can be exported or sent to live InteractiveAI instance.
- Agent
  - Actions can be exported as part of the trajectories.
  - Trained models can be saved as PKL files.
  - Training data is logged (e.g. rewards).



Visualization of the standard tree-observation



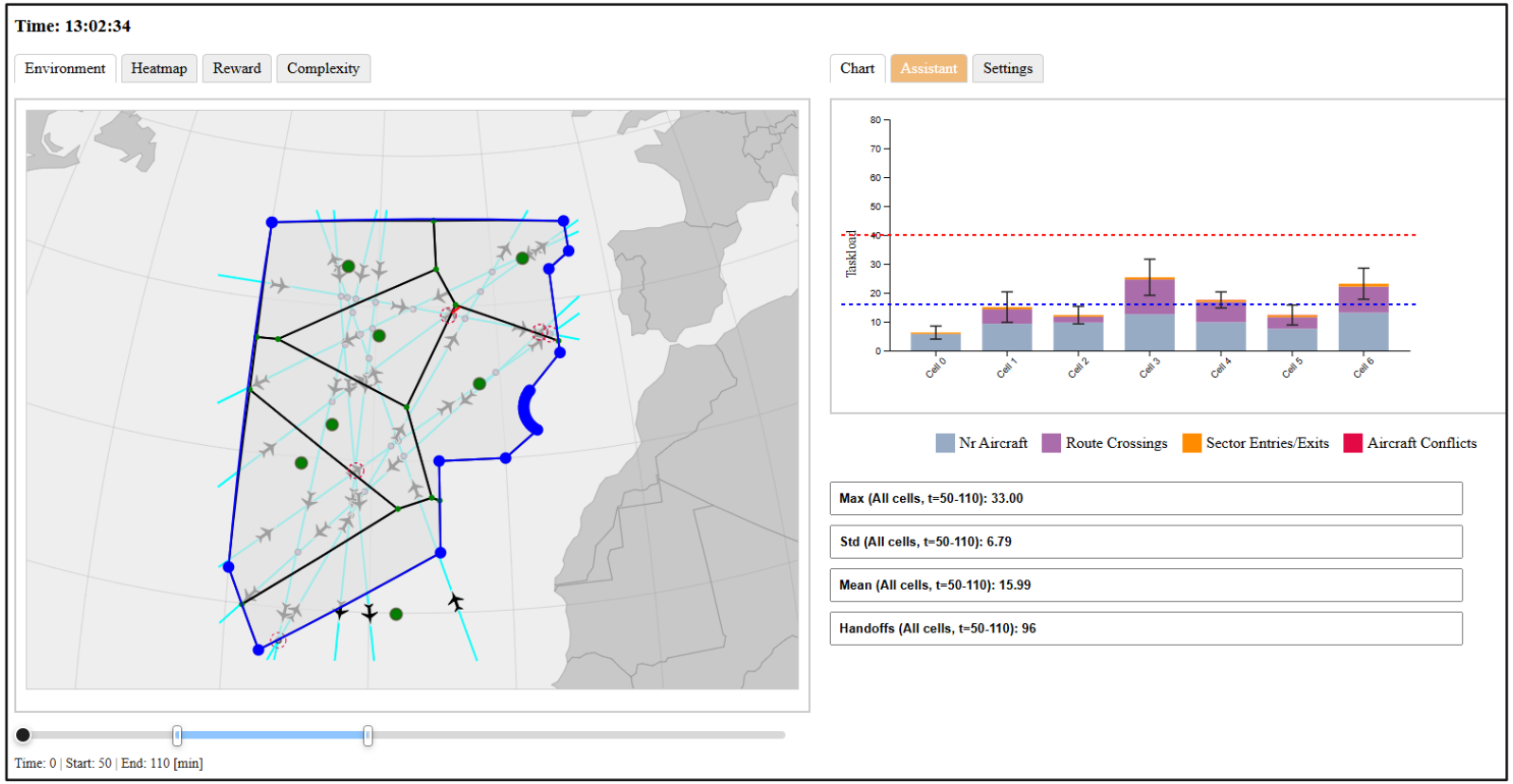
Real-world scenario demo (Olten, Switzerland): https://github.com/AI4REALNET/flatland-scenarios/raw/refs/heads/main/scenario_olten/img/olten_thumb.mp4

BlueSky: ATM Digital Environment

Developer: Delft University of Technology

ATM UC 1: sectorization

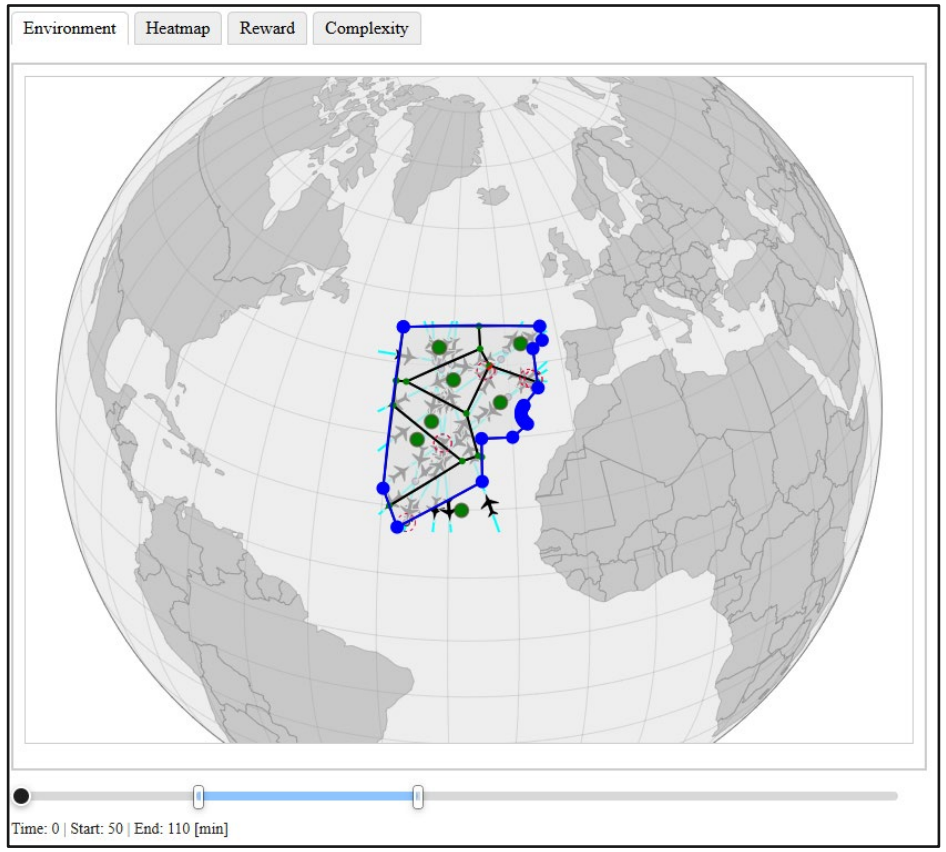
New Features



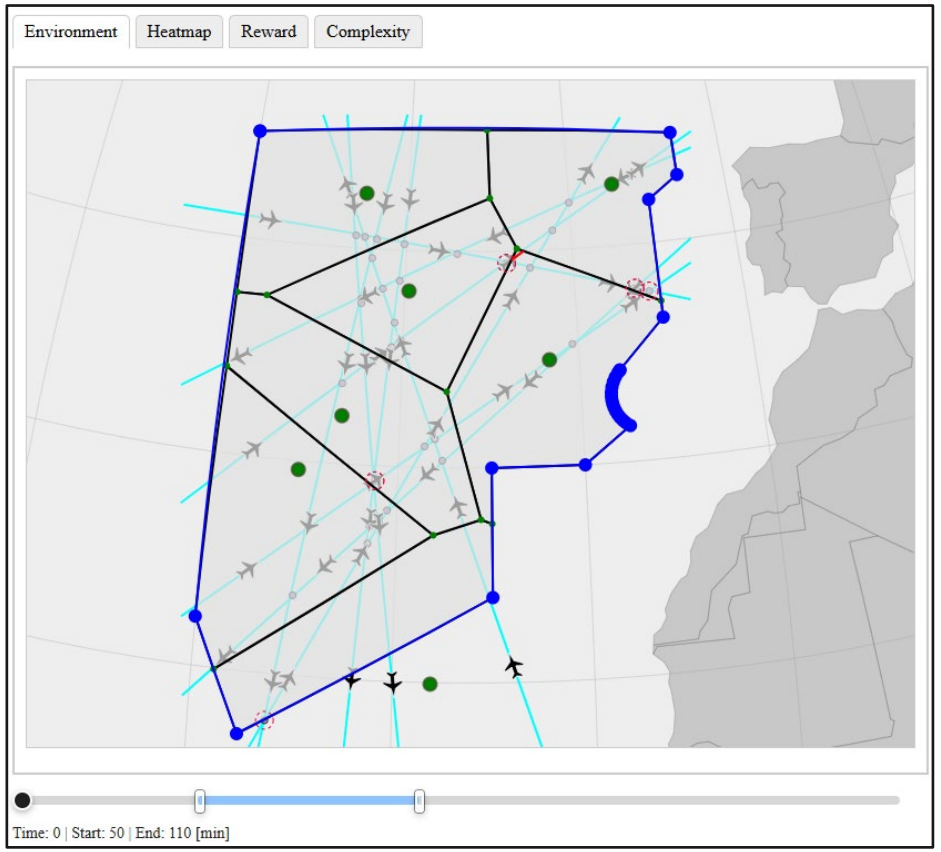
- ✓ Geospatial rendering on a globe
- ✓ Taskload (prediction) uncertainty
- ✓ Complexity view (taskload balance KPI)
- ✓ ATCo expertise assignment per sector
- ✓ Use aircraft icons & ATM text labels
- ✓ Experiment Manager
- ✓ Data logger
- ✓ Deployment on a webserver

Geospatial rendering for operational realism

Santa Maria FIR: ATM.UC1 environment for dynamic airspace sectorization.



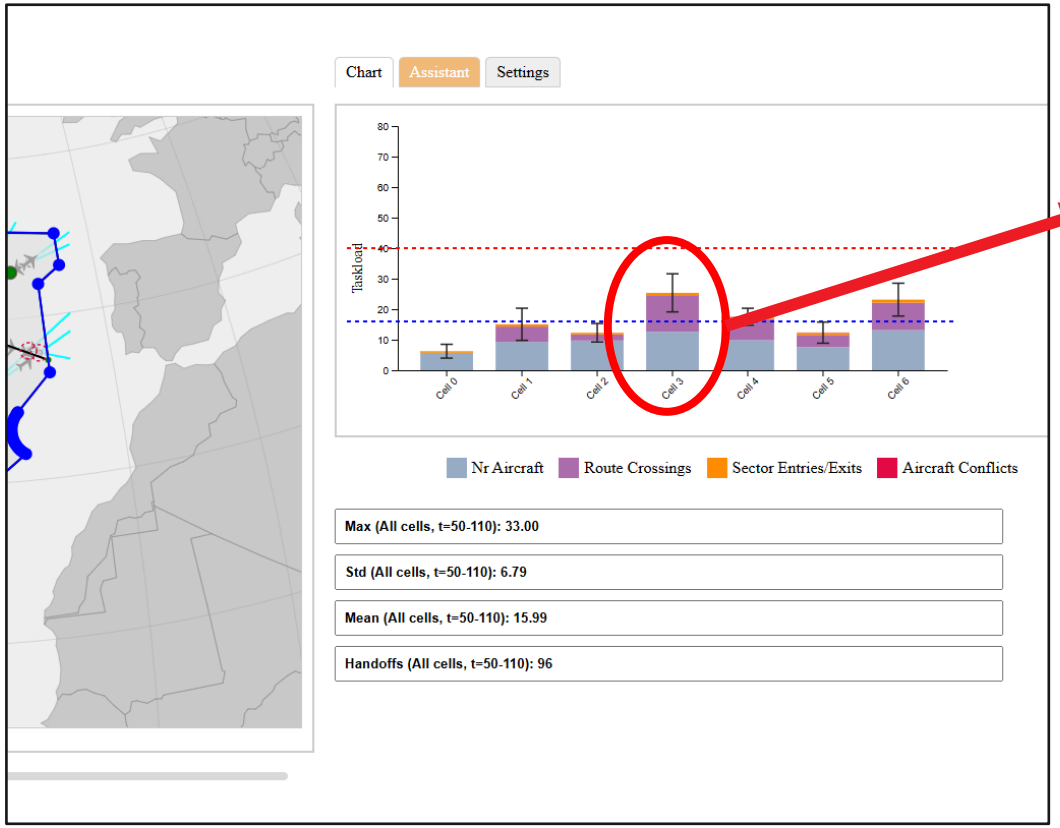
Zoomed out

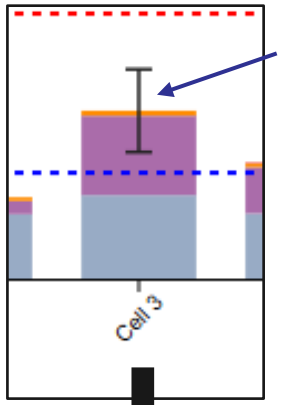


Zoomed in

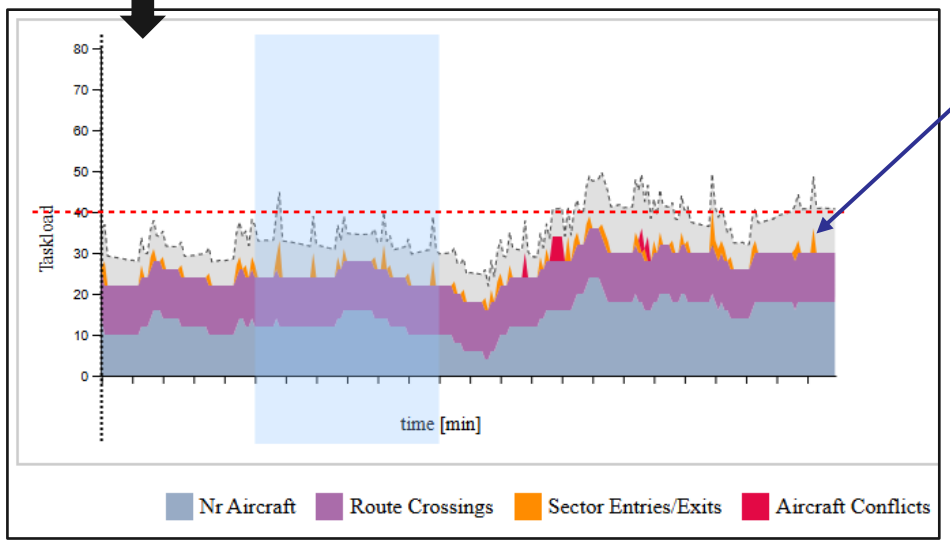
Geospatial rendering on a globe increases operational realism, enhancing overall user experience.

Taskload prediction uncertainty



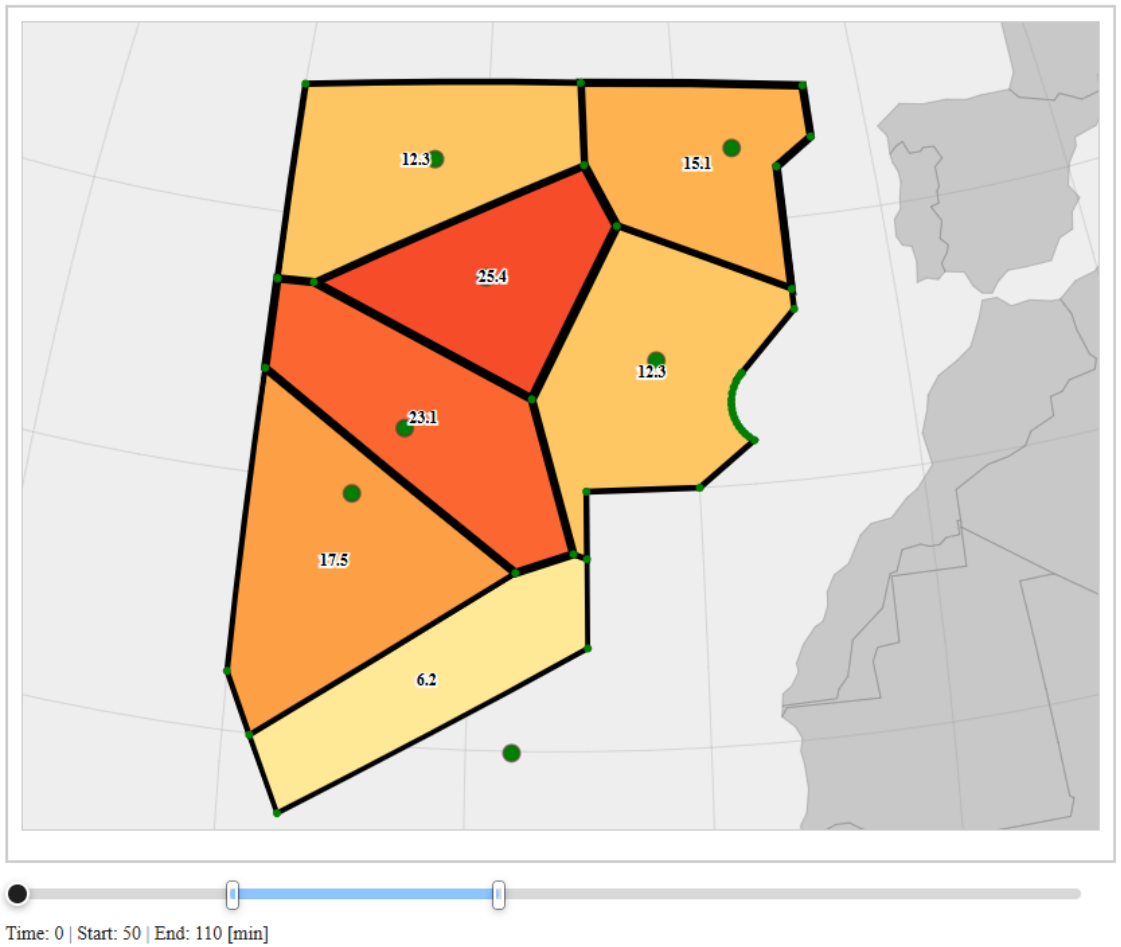


Error bars in the stack bar charts represent the level of uncertainty in predicted taskload per sector (i.e., polygon cell) within a given time horizon, stemming from uncertainty in aircraft trajectories.



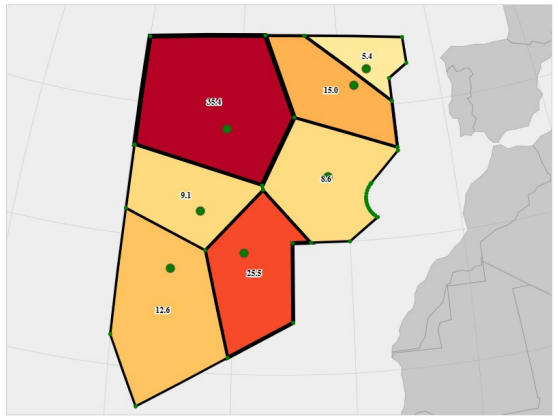
Grey areas visualized in the timeline plot of the predicted taskload indicate the worst-case taskload given modelled uncertainties.

Complexity view (taskload balance)

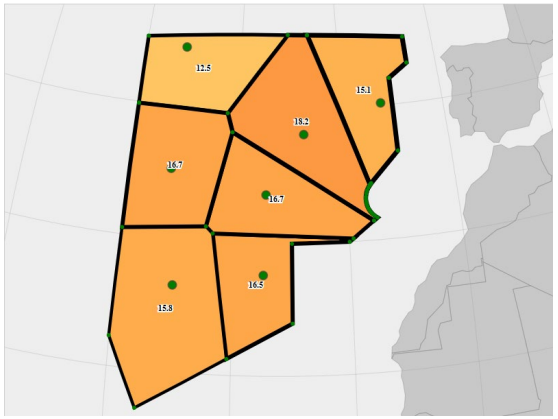


Color-coded polygons and **average complexity values** provide direct visual feedback on the taskload balance KPI. Maximum taskload can be configured and will be colored in dark red, with lowest predicted taskload near zero is colored light yellow.

The **line thickness** of each polygon represents the level of uncertainty in taskload balance: the thicker the line, the more uncertain its predicted taskload.

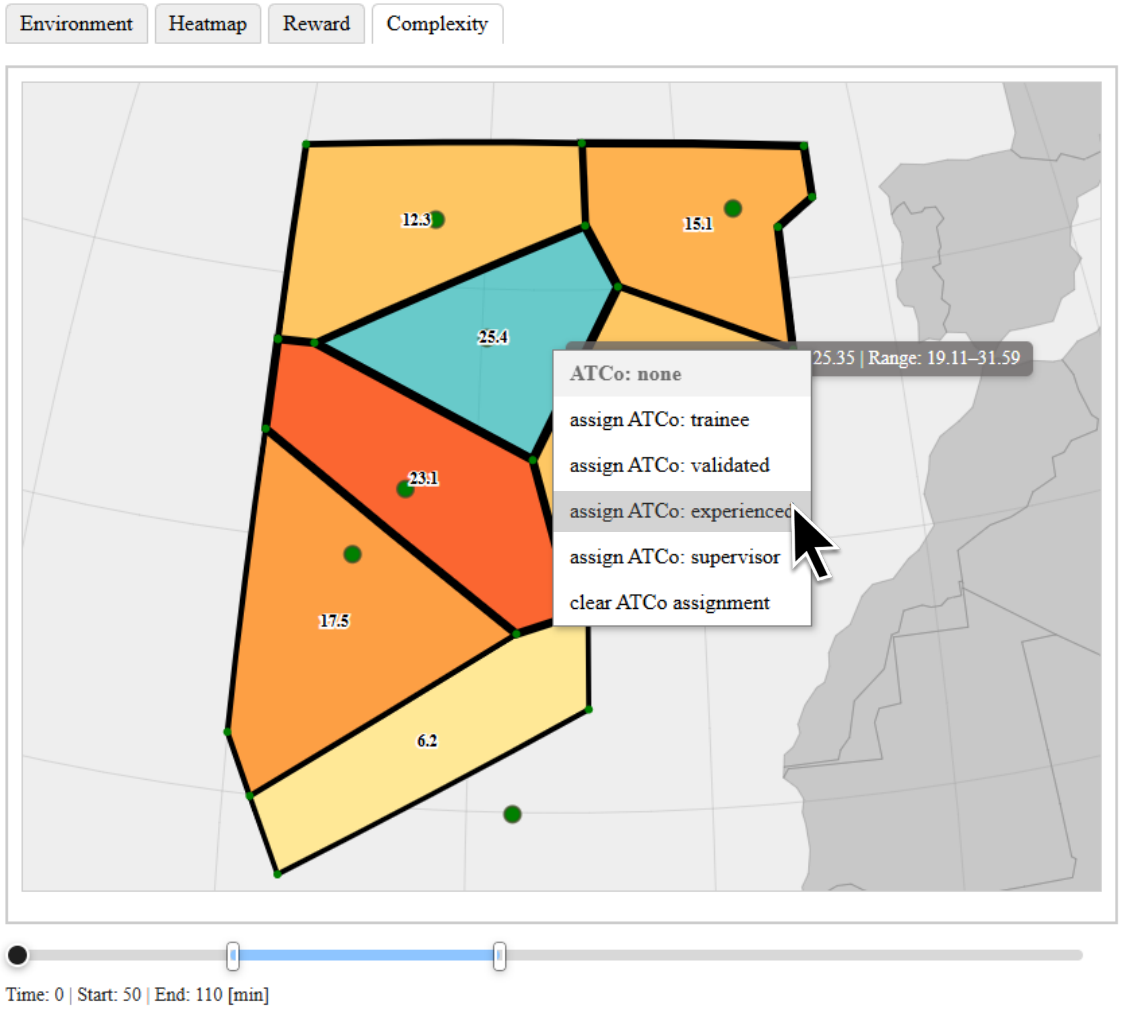


Unbalanced sectorization



Balanced sectorization

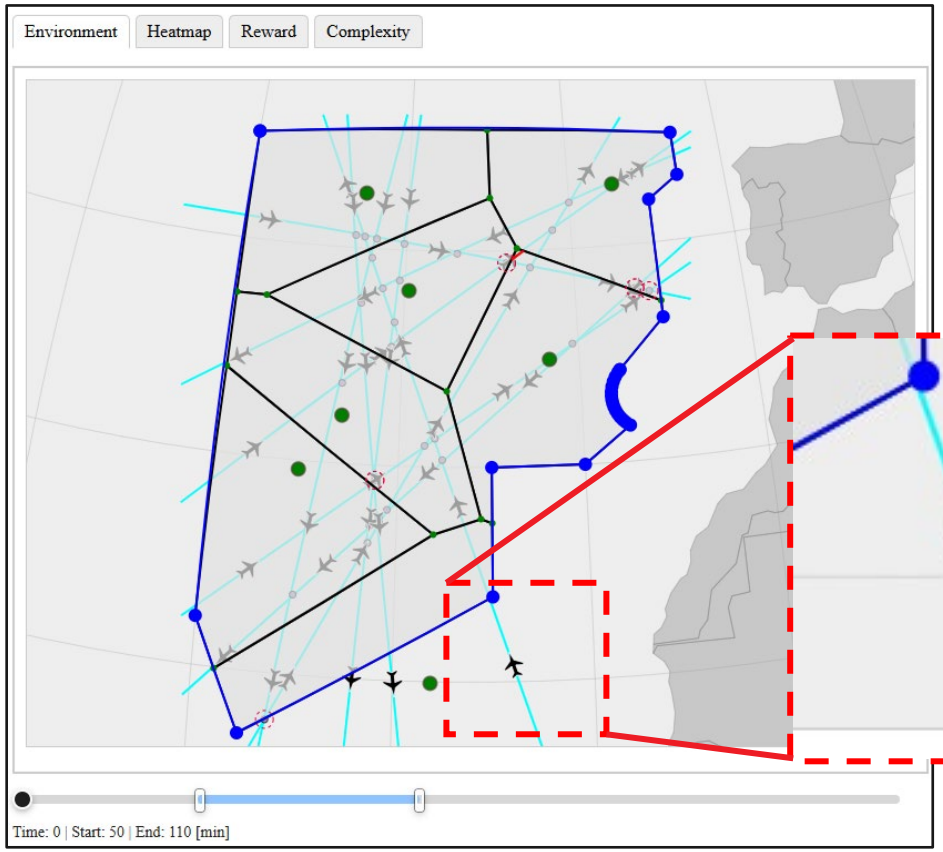
ATCo expertise assignment



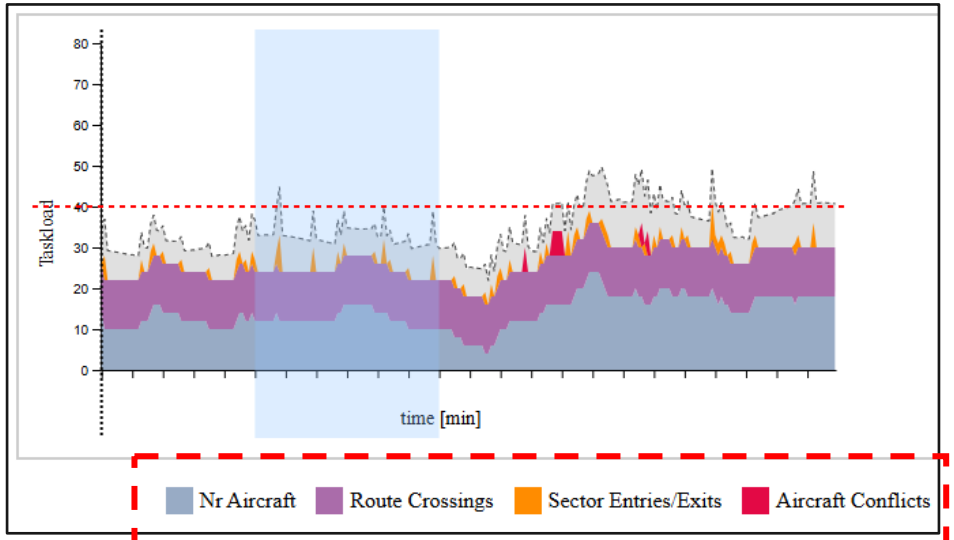
Enable supervisors to assign air traffic controller (ATCo) **expertise types** for handling a specific sector, given the sector's predicted taskload and uncertainty.

Although these labels are currently not used in the AI4REALNET sectorization assistant, collecting such data can facilitate collecting tacit operational knowledge to advance future algorithmic developments.

ATM-relevant icons and text labels

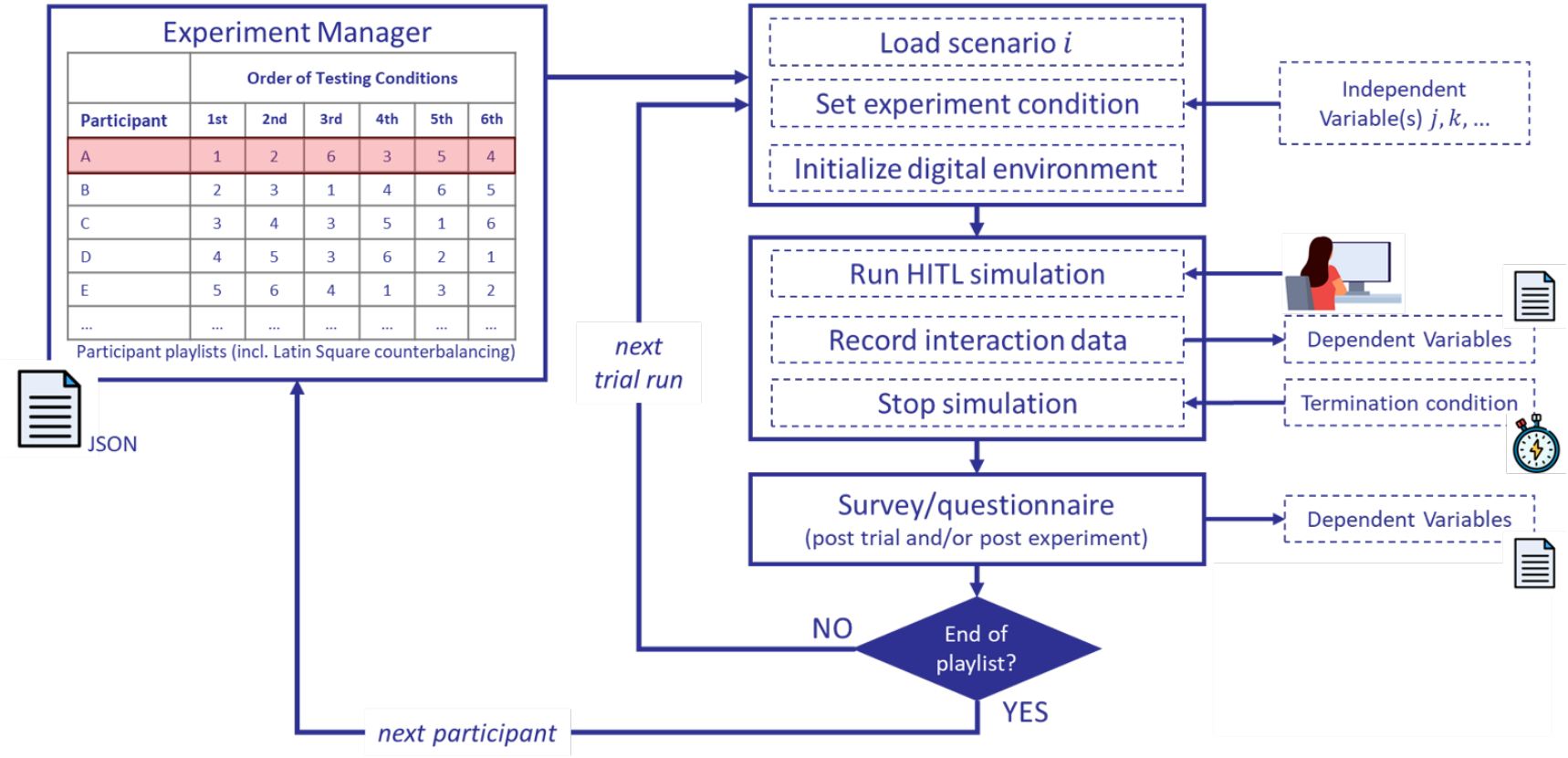


Using aircraft icons and ATM terminology in the text labels supports users' operational interpretations and adds meaning.



Nr Aircraft
Route Crossings
Sector Entries/Exits
Aircraft Conflicts

Experiment Manager



Experiment Manager and flow chart of its architecture needed for running experiments

An **Experiment Manager** and corresponding logic has been embedded in the ATMSectorization application that facilitates running self-paced experiments remotely within a web-browser. It loads personalized playlists defined in JSON format that specifies the sequence of scenarios and its configurations for the validation trials scheduled in WP4.

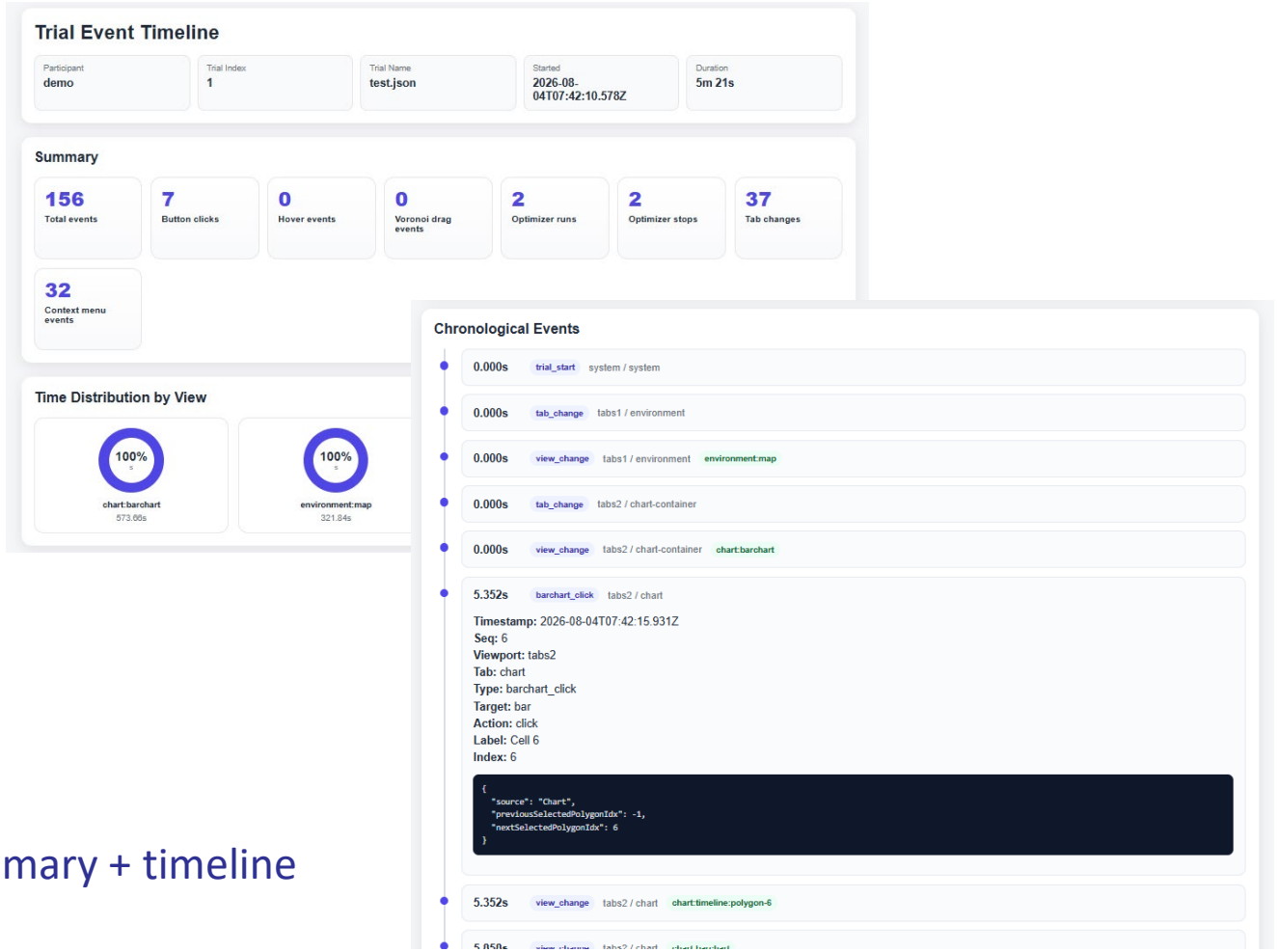
Data Logger

Data logging is an important feature to facilitate data collection and post-experiment analysis as part of WP4.

After each experiment trial, a ZIP is automatically downloaded containing all log files for later data analysis:

`trial_log_<ParticipantID>.zip`

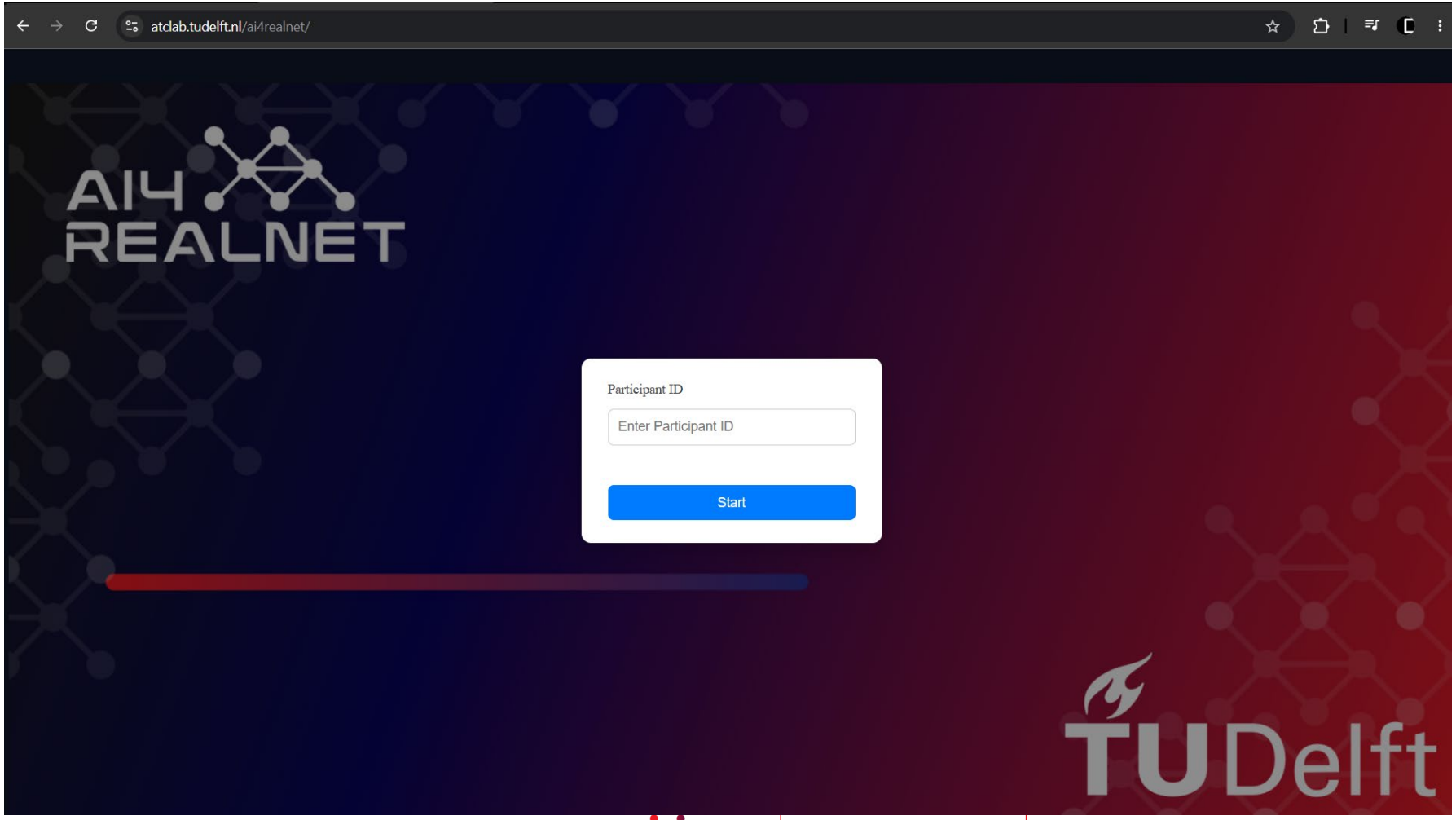
- `events.csv`
- `events.json`
- `trial_outcome.json`
- `trial_symmary.json`
- `survey_results.json`
- `timeline_report.html`



The HTML report contains an easy-to-read trial summary + timeline overview to facilitate JCF timeline analysis

Deployment on a webserver

https://atclab.tudelft.nl/ai4realnet/



edu.nl/typdc

The application has been deployed on a webserver, enabling people to remotely access it for participating in the experiment (using their unique ID) or for demonstration purposes (generic ID: **demo**).

ATM UC 2: path planning

BlueSky-Gym

Deploy-RL plug-in

TraffixGen

InteractiveAI connector

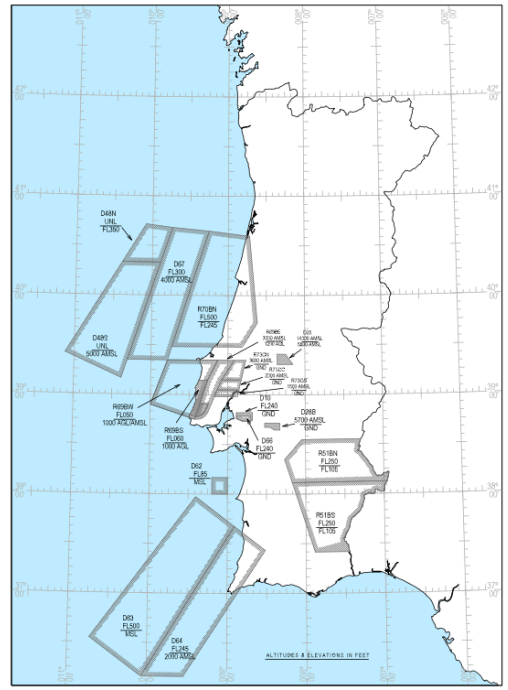
New Features



- ✓ Realistic Scenario generator
- ✓ Integration of BlueSky-Gym with the MORL-DOL library developed by UKASSEL
- ✓ Inclusion of evaluation metrics
- ✓ Added support for use case 2 specific features in InteractiveAI's codebase:
  - restricted areas and perturbations
  - protected zones around aircraft
- ✓ Build a UC2 connector between BlueSky and InteractiveAI to enable the interaction between the end user and AI agent:
  - Information is provided for aircraft leaving and entering the sector
  - Alerts for aircraft incursions in weather cells or volcanic ash cells
  - Alerts for aircraft incursions in restricted areas, and for loss of separation events

Context

- An RL-based decision support tool for stretegic path planning in Area Control Centres (ACC)
- Developed to support Air Traffic Control (ATC) and Flow Management Position (FMP) supervisors dealing with path planning complexities:
  - the activation of large military areas
  - the deterioration of weather conditions
- Demonstrated for the Lisbon Flight Information Region



Credits chart: NAV eAIP

https://ais.nav.pt/wp-content/uploads/AIS_Files/eAIP_Current/eAIP_Online/eAIP/html/eAIP/LP-ENR-6-en-PT.html

Training and Testing Architecture

Training architecture: BlueSky-Gym + MORL-DOL

- BlueSky-Gym (release 1.2 and 1.3) provides a set of simplified scenarios, procedurally generated, which are used to train models with reinforcement learning agents (custom or from standard libraries).
- MORL + Deep Optimistic Linear Support library is connected to BlueSky-Gym (this release) by exposing the reward components representing different objectives.

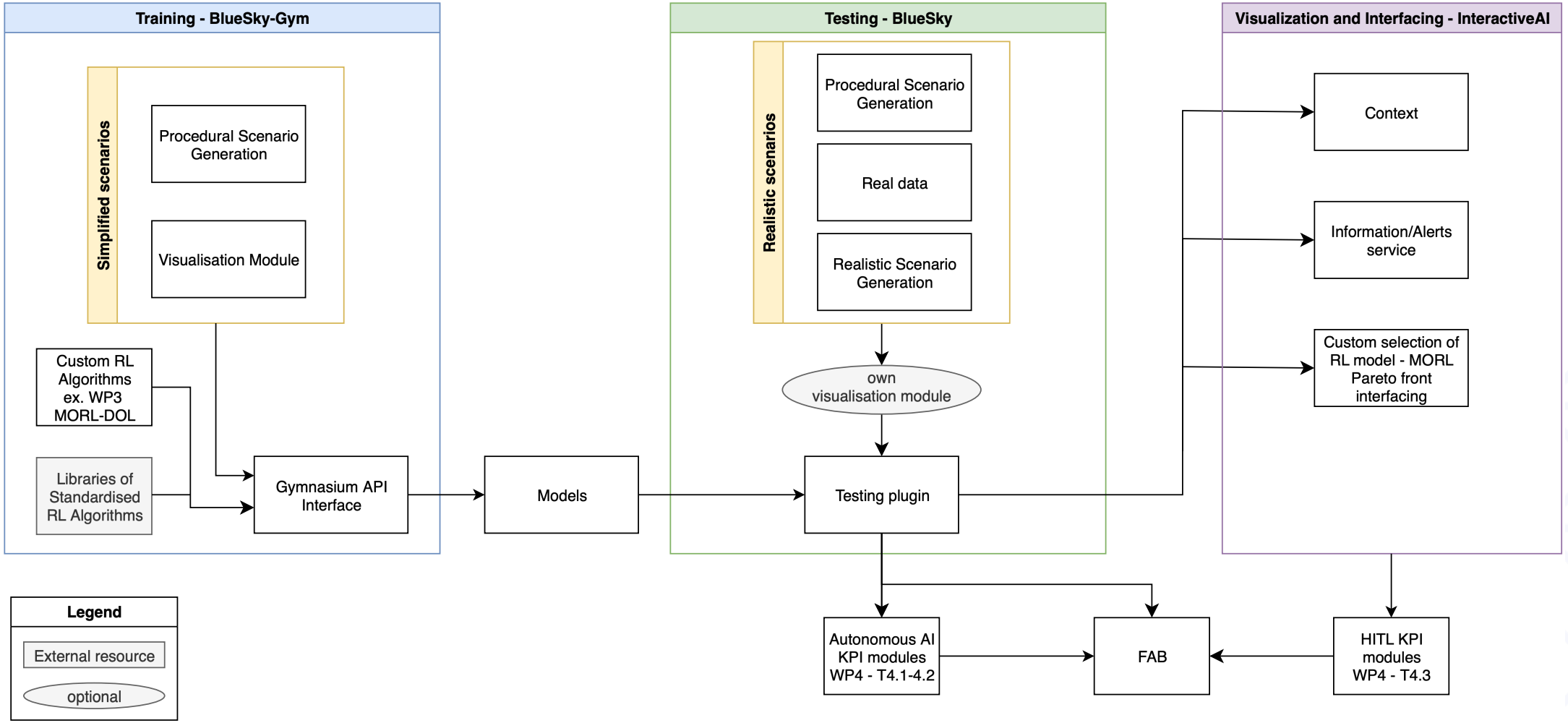
Testing architecture – autonomous RL: BlueSky plugins (synthetic/realistic scenario generation, perturbation generator and RL deployment) + FAB

- The BlueSky plugins (release 1.3) loads sector data and generates synthetic aircraft, restricted areas and perturbations. The RL model deployment sends observations to the previously trained AI models from the BlueSky-Gym environments, which in turn control the speed and heading of the aircraft.
- The BlueSky engine measures KPIs through simulations (this release), and sends them to the FAB platform connected through an API (release 1.3).

Decision support tool: BlueSky plugins + InteractiveAI

- A connector (this release) is built to bridge the BlueSky plugins for deployment of the RL models on generated scenarios with InteractiveAI.
- Alerts and Information are provided to the user in the dedicated tabs (this release).
- Interaction with a Pareto front interface allow the user to custom pick a model to run on specific aircraft.

Full Architecture



TraffixGen - a realistic scenario generation

- Python package for processing and analyzing flight traffic data
- Classify and analyze flights based on flight type or other characteristics
- BlueSky plugin used to generate realistic Origin–Destination (OD) distributions
- Capability to add jitter to scenarios, to simulate delays
- Interaction through a graphical user interface

Training stage: BlueSky-Gym general description



A Gymnasium style library for standardized Single-Agent Reinforcement Learning research in Air Traffic Management

- A platform to accelerate RL research in ATM, bridging gap between fundamental research and full-scale experiments
- Fully compatible with standard Gymnasium Reinforcement Learning API
- Compatible with RL libraries (StableBaselines-3, RLLib), allows for own algorithms
- Developed in Python

Environments general description

- Scenarios are simplified, but representative of full-scale problems
- Customizable complexity and traffic types thanks to a modular design, providing standard observation and action spaces, reward functions for:
  - Horizontal environments for lateral navigation (observations: drift and distance from destination. Distance, bearing and speed with respect to intruders; actions: heading changes; reward: intrusions, drift, destination reached)
  - Vertical environments for vertical navigation (observations: altitude, vertical speed, target distance, runway distance. Bearing, speed and altitude with respect to intruders; actions: intrusion and target altitude distance, crash penalty)

Training stage: available environments

Environments:

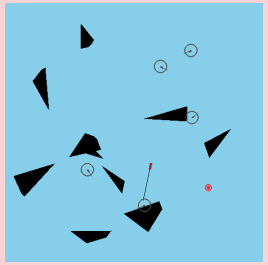
- StaticObstacleCREnv-v1
- StaticObstacleSectorEnv-v0
- StaticObstacleSectorCREnv-v1 -> Integrated with the MORL library
- CentralisedStaticObstacleEnv-v0
- CentralisedStaticObstacleCREnv-v1
- CentralisedStaticObstacleSectorCREnv-v1

https://github.com/AI4REALNET/bluesky-gym/blob/main/bluesky_gym/envs/static_obstacle_cr_env.py

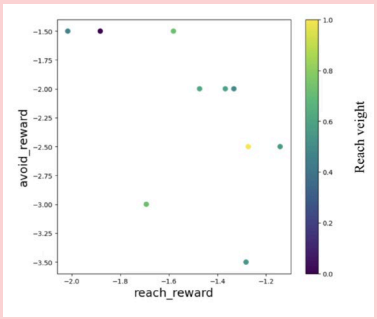
Multi-Objective Reinforcement Learning (MORL):
integration with BlueSky-Gym training and testing

Training architecture
(BlueSky-Gym + MORL-DOL)

MORL trains multiple models for differently weighted reward components:



StaticObstacleCR env

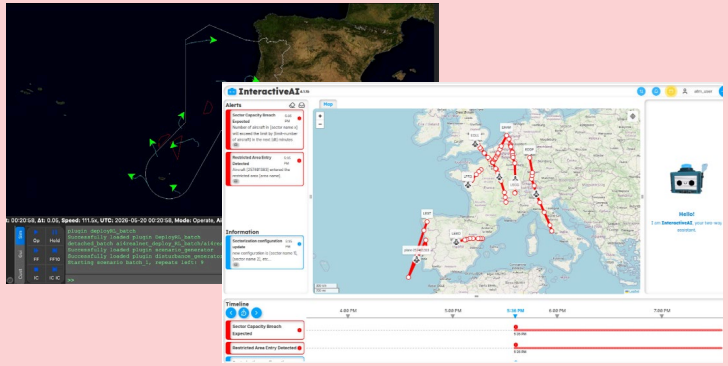


w1 (safety) + w2 (efficiency)

output

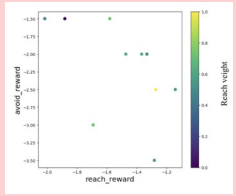
$$\begin{cases} \text{MODEL}(w_1', w_2') \\ \text{MODEL}(w_1'', w_2'') \\ \ldots \\ \text{MODEL}(w_1^n, w_2^n) \end{cases}$$

Testing architecture
(BlueSky + InteractiveAI + MORL interface)



BlueSky (backend) runs a default model from the training output.
InteractiveAI shows the resulting trajectories and a Pareto front interface





Human selects a design point per aircraft: a different model is loaded in place of the default

output → Interaction data

Inclusion of evaluation metrics

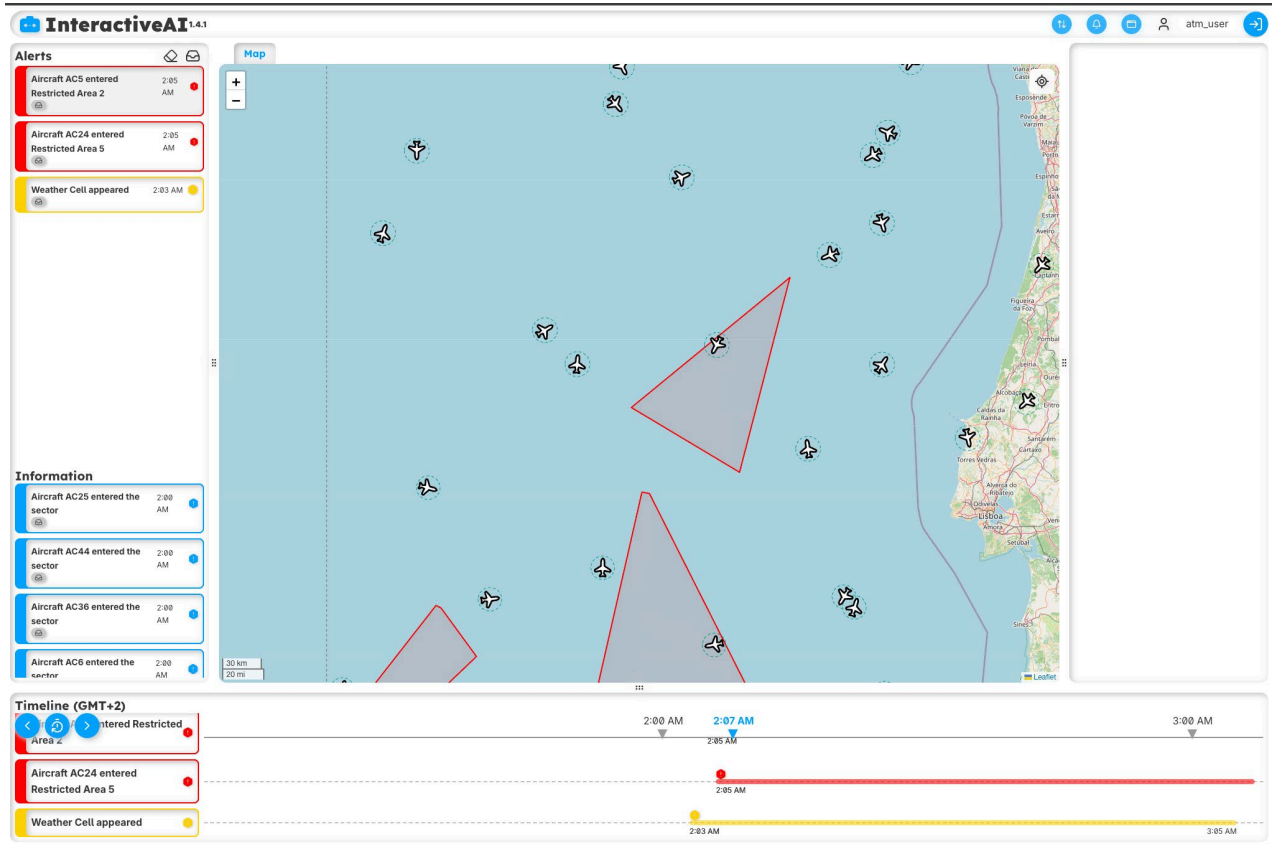
- Intrusions (total intrusion time + number of intrusions events)
- Separate reward components (drift, destination reached, intrusions, crashed)
- Minimum closest point of approach (with respect to intruders and restricted areas)
- Number of obstacles encountered (<20 NM), number of other aircraft encountered (<20 NM)
- Additional path length (compared to straight distance or distance computed by a path planning algorithm), ratio (actual/planned) + additional time
- Action (heading and speed delta): amount of reversals, rate of saturation, average value and average distance from action at the previous timestep

The metrics are available for the StaticObstacleCR environment:
https://github.com/AI4REALNET/bluesky-gym/blob/metrics/bluesky_gym/envs/static_obstacle_cr_env.py

Integration of UC2 in InteractiveAI

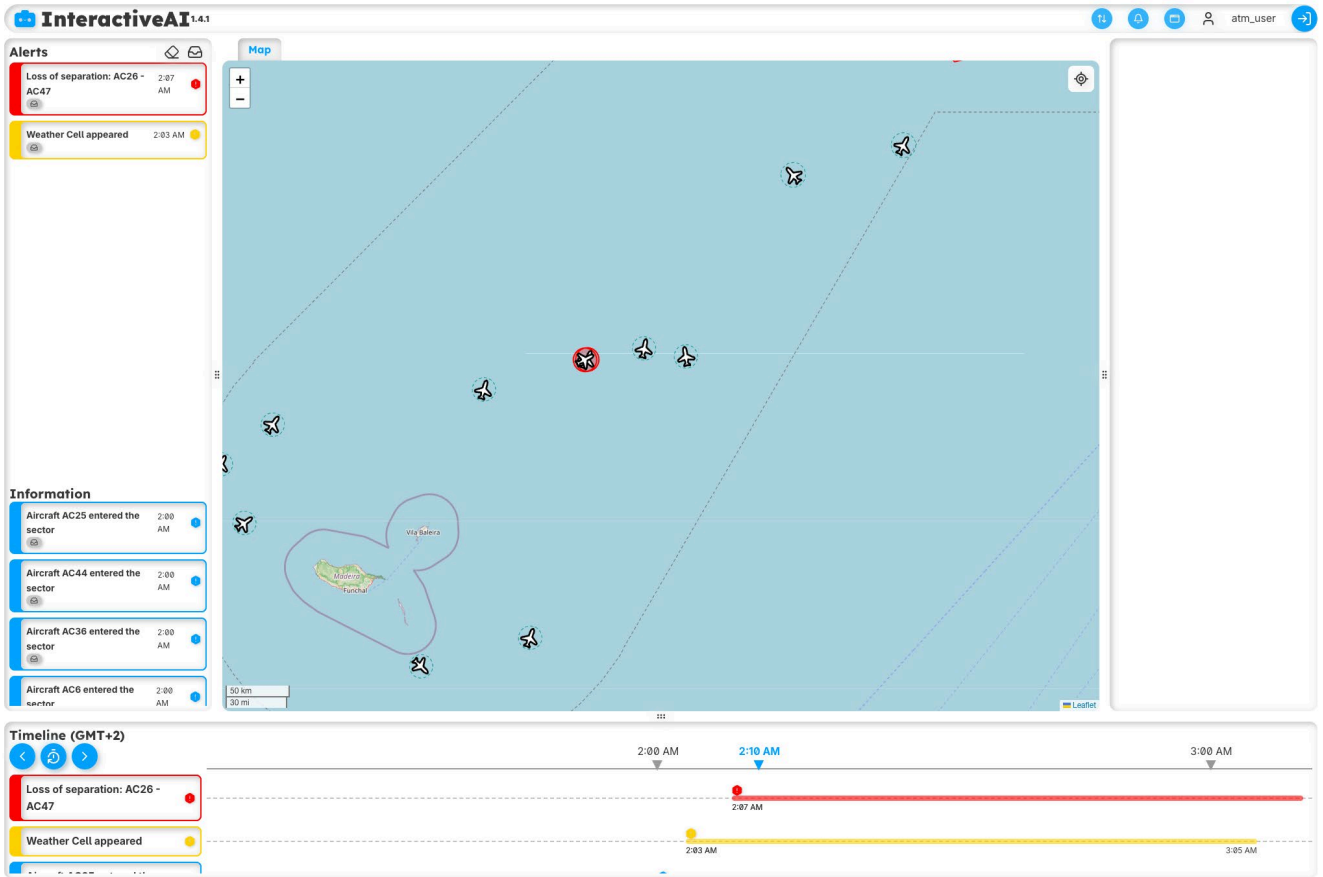
- BlueSky is the engine of the simulations:
  - Used to simulate scenarios, synthetic or historical
  - The aircraft are controlled by an RL agent via BlueSky's plugins (ai4realnet_deploy_RL_batch.py, see Deliverable 1.3).
- InteractiveAI is the frontend interface with which the FMPs interact.
  - The state of the simulation is pushed as context to InteractiveAI
  - Information is displayed to the user: entry/exit of aircraft, weather or volcanic perturbation occurrence
  - Events (such as the occurrence of losses of separation or the incursions in restricted areas) trigger notifications that are displayed to the user, with various degrees of urgency.

Restricted areas



- Restricted areas are shown in red
- All areas are associated with a name, visible when hovering over the areas.
- High priority Alerts are shown if an aircraft intrudes a restricted area

Protected Zones

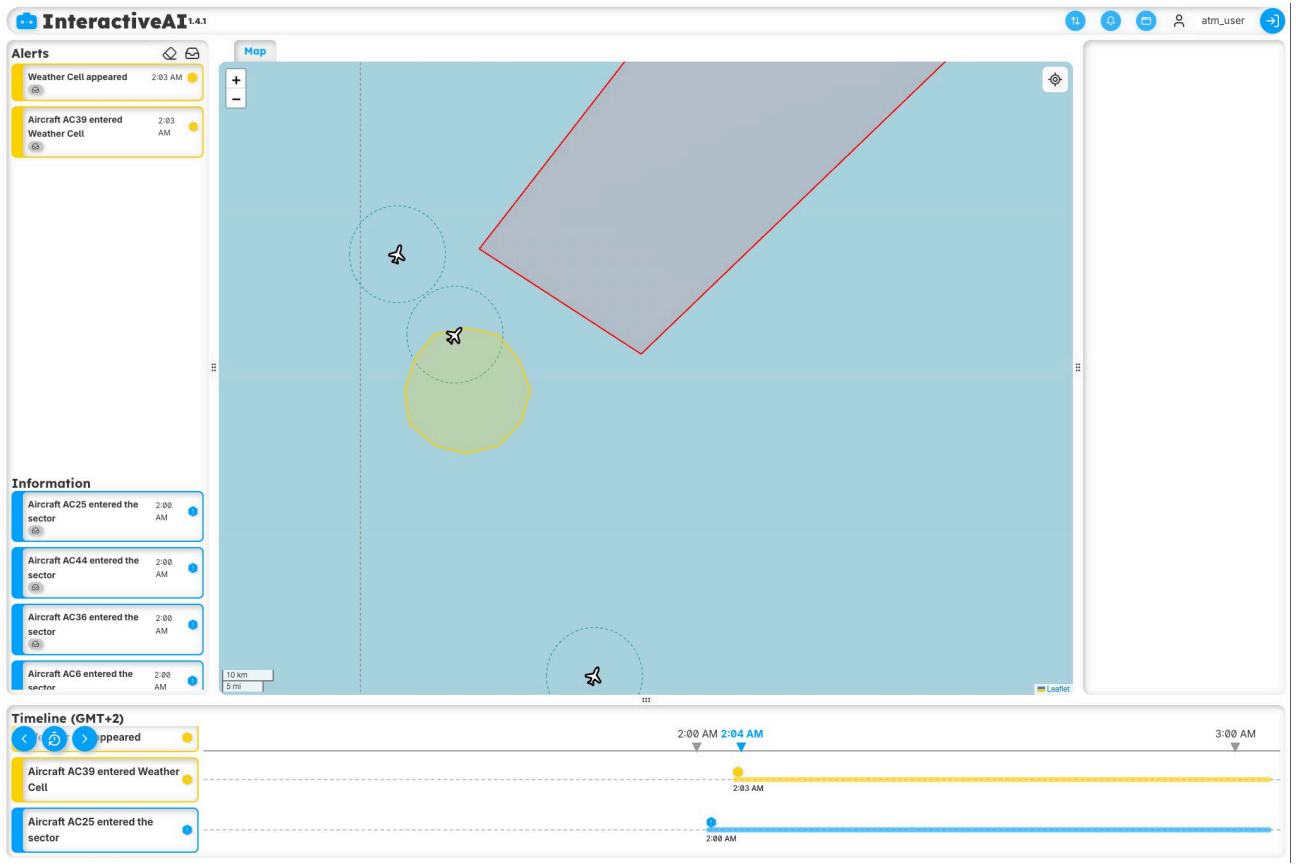


Aircraft are required to maintain a 5NM distance from each other for safety. The areas with 5 NM radius centered around each aircraft are protected zones.

The presence of an aircraft within another aircraft protected zone is a loss of separations.

- Protected zones are displayed in green in the interface, if no aircraft is intruding.
- In case of loss of separation:
  - High priority Alerts are issued if two aircraft invade each other's protected zones.
  - Protected zones are colored red to immediately draw the controller's attention.
  - Moving the cursor on the aircraft displays the aircraft ID.

Weather/volcanic perturbations



- Weather and volcanic perturbations are shown in yellow
- All areas are associated with a name, visible when hovering over the areas.
- Medium priority Alerts are shown if a weather/volcanic ash cell appears (as driven by BlueSky's perturbation generation module).
- The approximate duration of the event is displayed on the timeline (to mirror the ATCO's workflow when weather forecast data would be available in place of the generated perturbations).
- Information notifications are provided when the perturbations are cleared.
- Medium priority Alerts are shown if an aircraft intrudes either a weather/volcanic perturbation cell

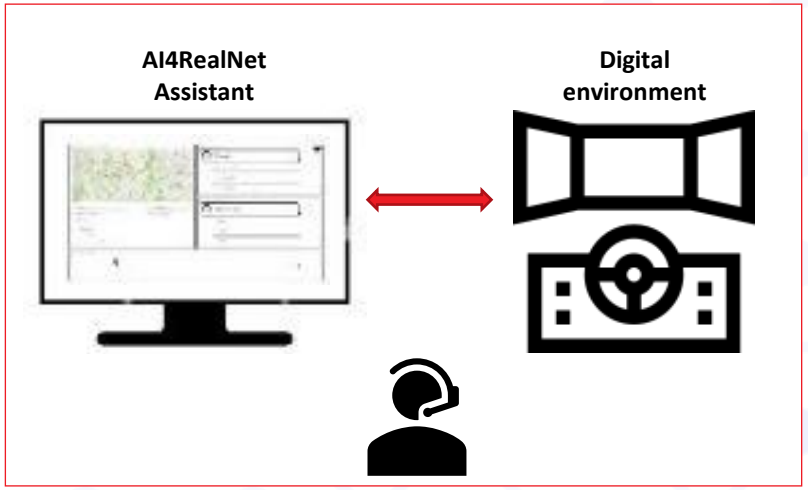
InteractiveAI: AI4REALNET Framework

Development Lead: IRT SystemX

Contributors: EnliteAI, FHNW, Fraunhofer, InescTec, EnliteAI, UKassel

InteractiveAI Platform - Objectives

- The **InteractiveAI** platform is a **generic** technical base used to build interactive assistants for different industrial sectors
- The assistant objectives:
  - Display a user-friendly context view about the digital environment : real time state
  - Notify the operator in case of incidents
  - Allow the operator to interact with his environment
  - Ask for help
  - Display recommendations proposed by AI agents
  - Give feedback on AI outputs

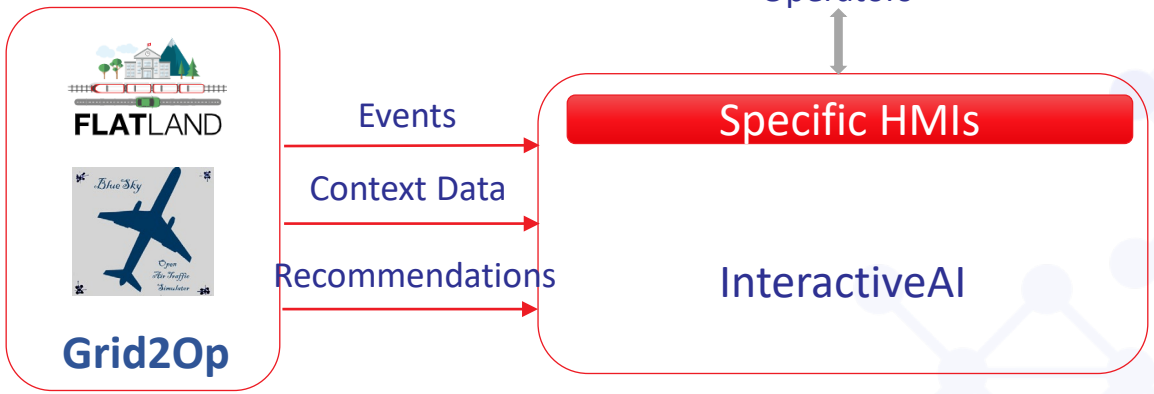


InteractiveAI Platform - Description

- **An interactive AI Assistant Platform for Real Time operations**
  - A generic technical base for all the industrial sector
  - Specific HMIs for each application domain
  - Communication with the digital environments
  - Facilitate the interaction of the operators with different environments



Operators



Digital environments

InteractiveAI
Github repository

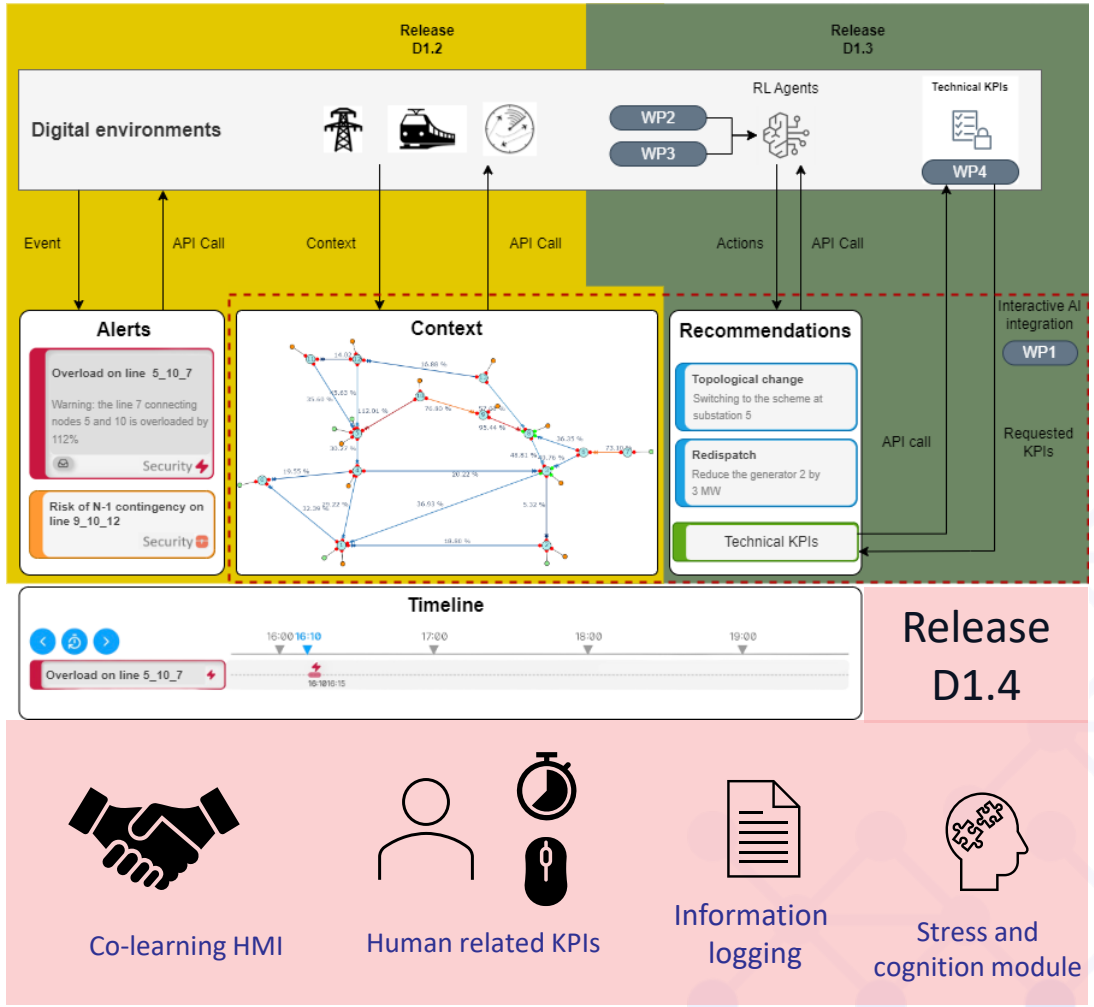
Inputs

- Generic InteractiveAI Platform : Open source
- Different use case scenarios
- Specific needs for HMIs
- Generic API connectors specifications
- Trained AI agents for each domain

Release plan and objectives

- **Release D1.2 (submitted)**
  - Digital environment enhancement
  - Integration with Interactive AI
- **Release D1.3 (submitted)**
  - Interoperability connectors
  - KPIs
- **Release D1.4 (This release)**

  Full ecosystem: environment & benchmarks
  - Human related KPIs
  - Information logging
  - Stress and cognition module
  - Human experience questionnaires
  - Co-learning HMI
  - Agent as a service (A3S)
  - AI Agents
    - CurriculumAgent
    - ExpertAgent
    - SoftGNN
    - Multi-objective RL
  - Uncertainty Quantification



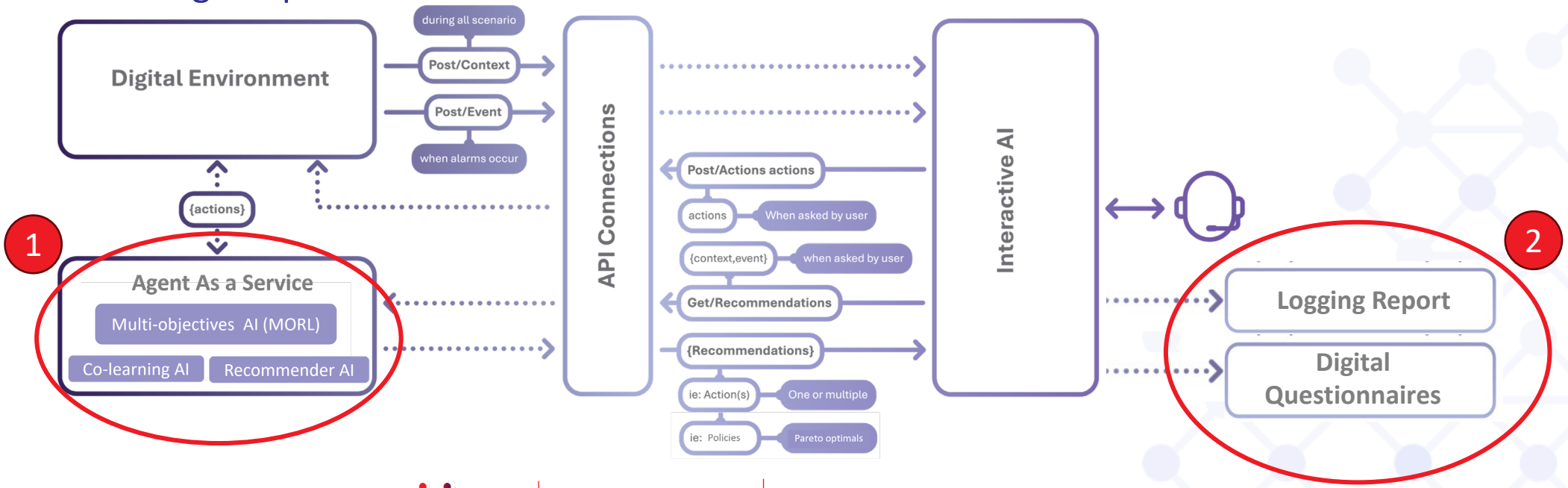
Generic adaptations – InteractiveAI- Final Release

1 AI Agents & Interaction Support

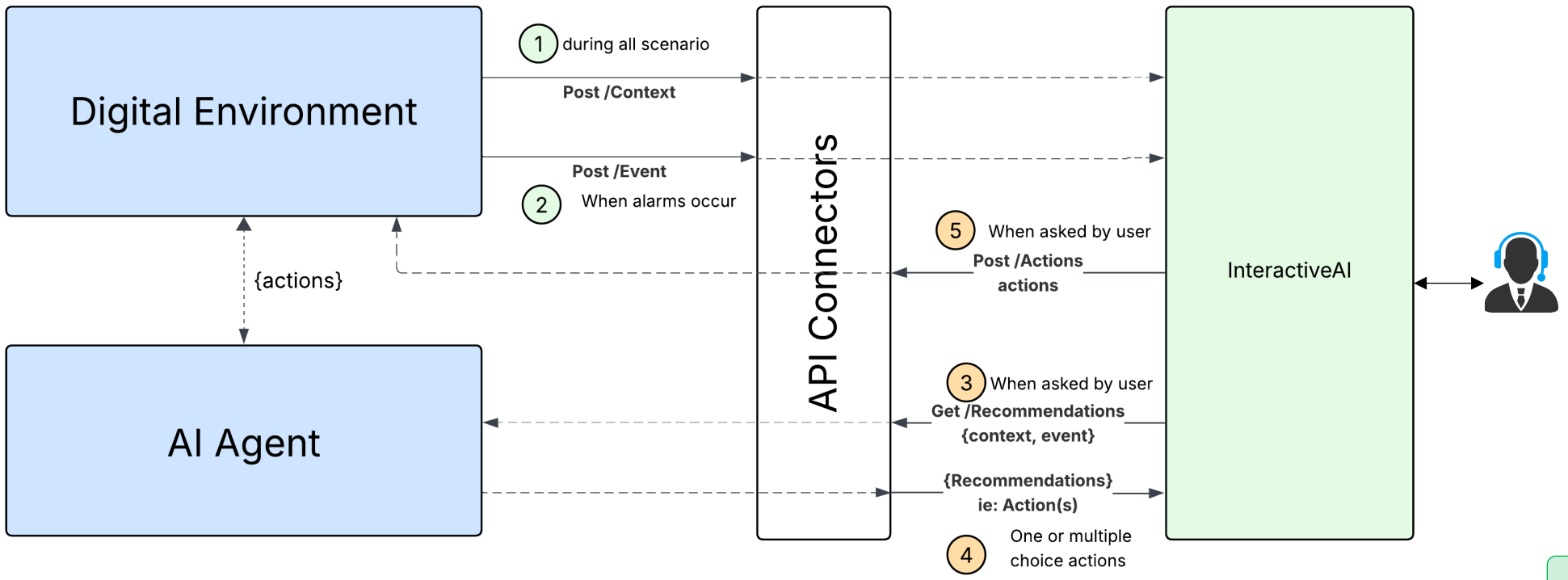
- **Agent As a Service / Multi-Agent support** (Directive AI, Multi-Objective AI ,Co-Learning)

2 User Study Support

- Logging service → session reports & KPIs
- Customizable digital questionnaires



Block diagram



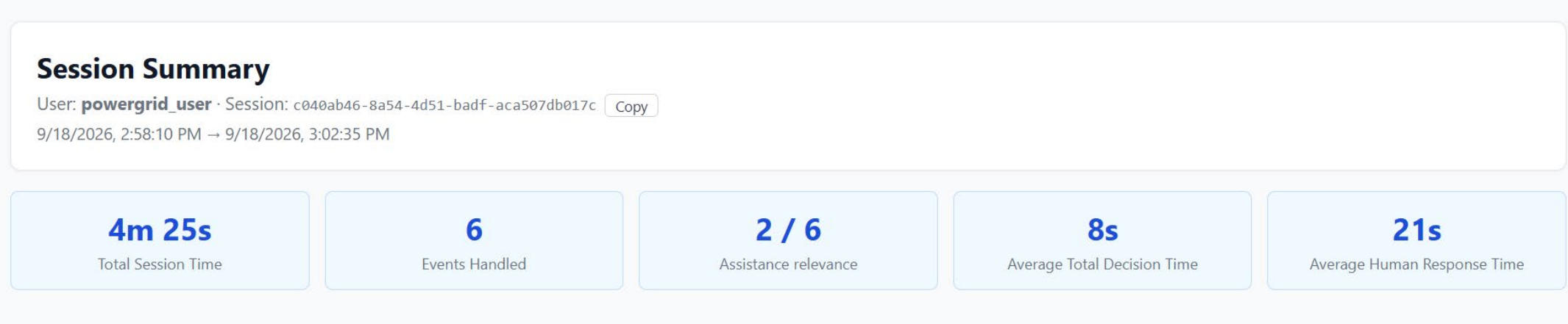
Release#1

Release#2

Human related KPIs

- **During the session, InteractiveAI captures the following human interaction KPIs:**
  - **Total decision time:** total time taken to resolve an issue
  - **Human response time:** time taken by the human operator to make a decision
  - **Assistance relevance:** number of times the operator considers the AI output
  - **Total session time:** total duration of the session

Per session

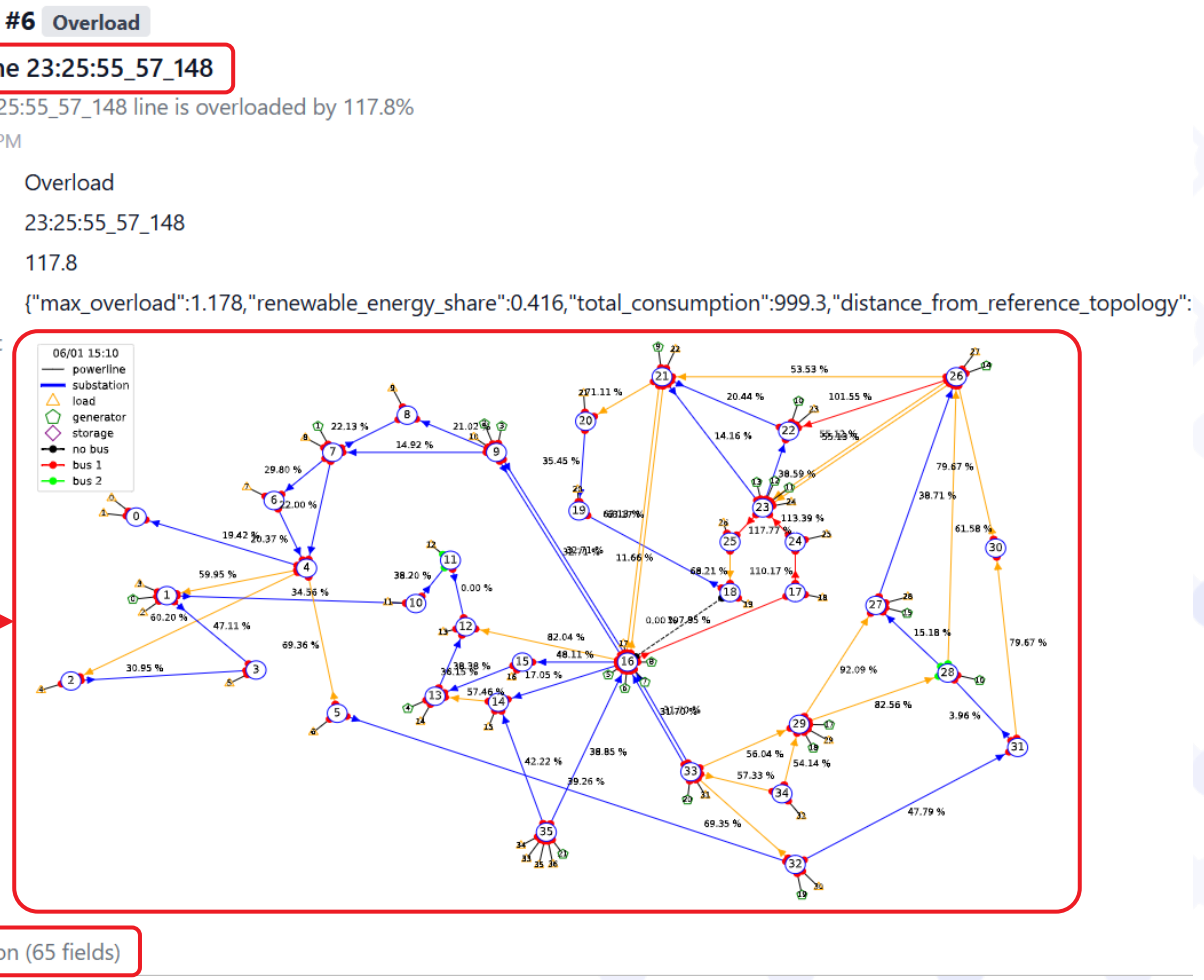


Per event



Information logging

- **InteractiveAI provides a detailed report of the environment status and operator actions, with timestamps (json file and HTML report)**
  - **Alerts:** alerts that occurred during the session
  - **Environment context:** context of the environment when an alert occurs
  - **Help requests:** when the operator asks for assistance
  - **AI outputs:** recommendations or outputs provided by the AI
  - **Operator decisions:** decisions made by the operator
  - **Operator feedback:** feedback provided by the operator



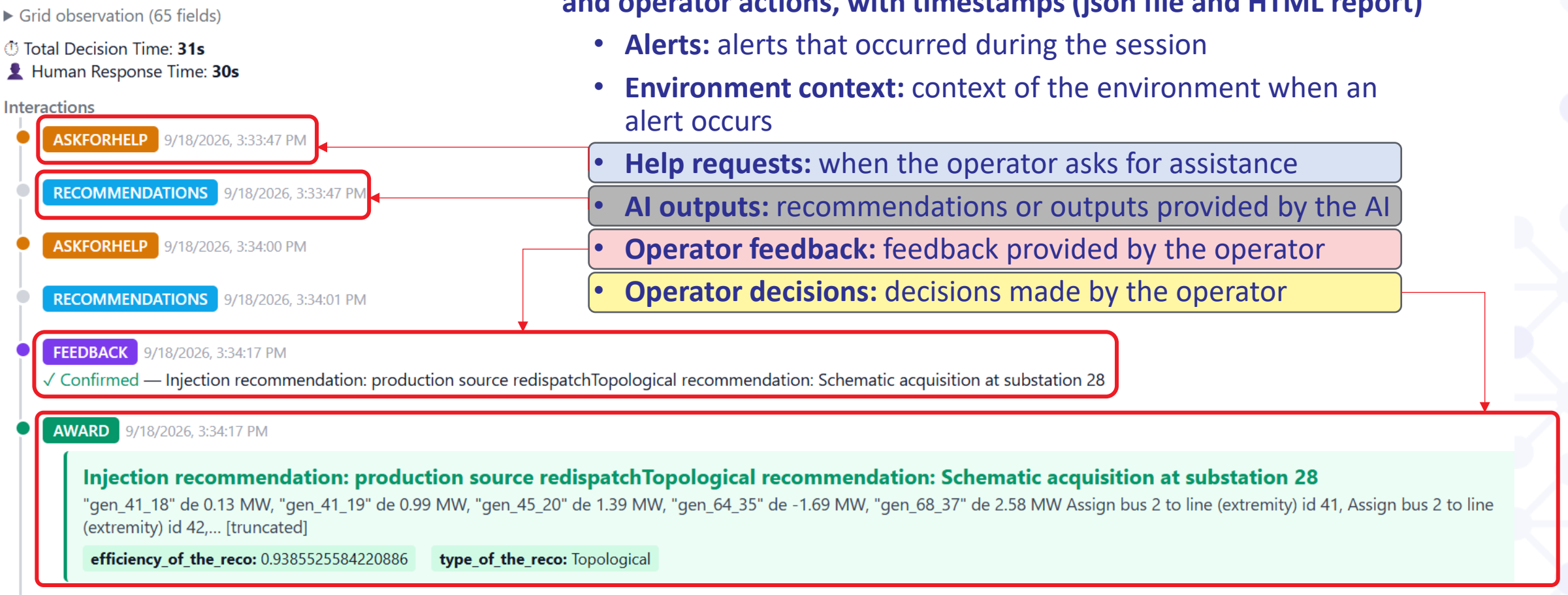
Exemples

historic-session-powergrid_user-2026-09-17T13-38-11-775Z.html

historic-session-powergrid_user-2026-09-18T12-58-10-316Z.json

Information logging

- **InteractiveAI provides a detailed report of the environment status and operator actions, with timestamps (json file and HTML report)**
  - **Alerts:** alerts that occurred during the session
  - **Environment context:** context of the environment when an alert occurs
  - **Help requests:** when the operator asks for assistance
  - **AI outputs:** recommendations or outputs provided by the AI
  - **Operator feedback:** feedback provided by the operator
  - **Operator decisions:** decisions made by the operator



Stress and cognition module

Definition

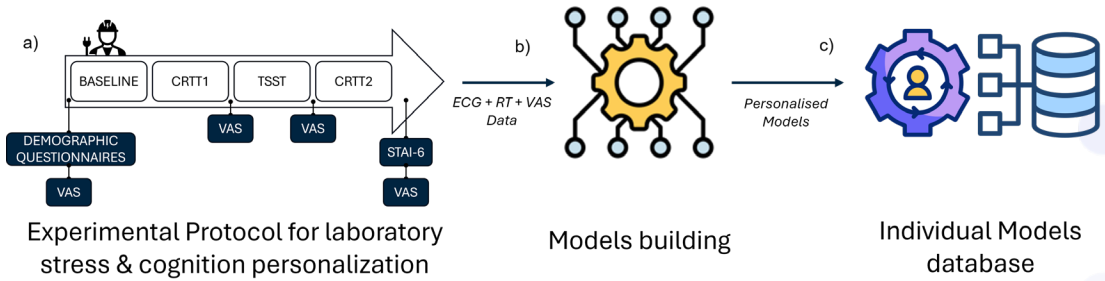
The **Human Assessment Model** (**HAM**) is an Artificial Intelligence (AI) based procedure for the continuous, real-time evaluation of cognitive load and stress in high-risk operators using physiological signals.

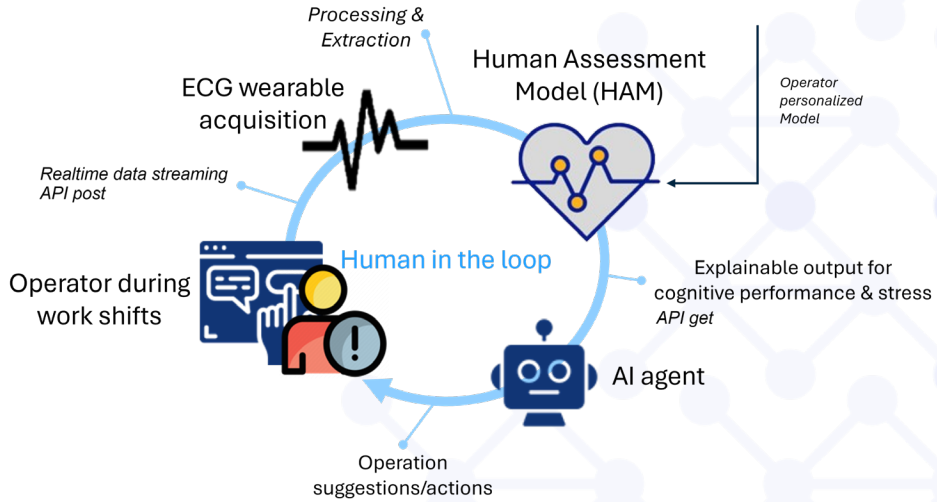
Motivation

Physiological signals are better suited for real-time scenarios and allow for a stronger interaction between human operators and AI agents to improve suggestion mechanisms.

Implementation

The HAM consists of a 2-step procedure: an offline phase to characterize the stress response of an individual; an online phase to evaluate cognitive load and stress level during work shift with a human-in-the-loop approach.





Application domain

Stress and cognition module

Personalization

An experimental paradigm is proposed to analyse cognitive performance before and after applying a social stressor while acquiring electrocardiographic (ECG) data through a wearable sensor.

After signal processing, feature selection and hyperparameters tuning are carried out to train participant-specific models.

Models are stored in a database to be retrieved and loaded through API requests.

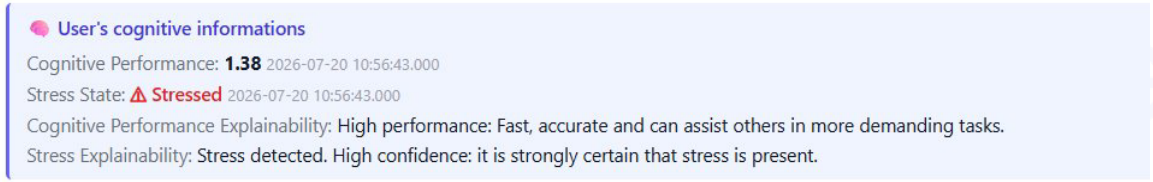
Real-time usage

ECG data are collected during operators' normal work shifts while interactive with AI agents

Data is transferred to check for signal quality and feature computation during the operators' decision making process.

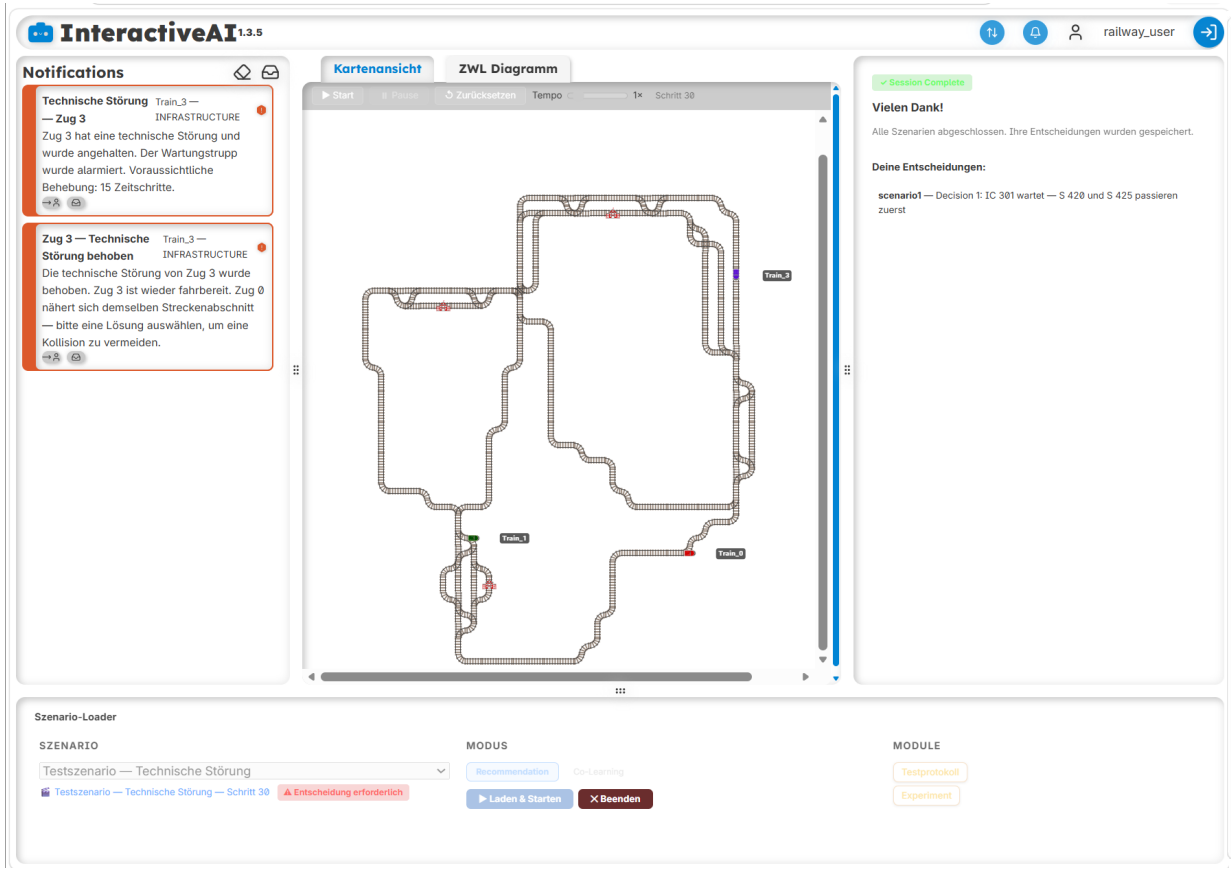
Models outputs are converted into strings for interpretability and retrieved by InteractiveAI using a dedicated API.

AI agents receive these operator data jointly with operational information for output human-based personalization.

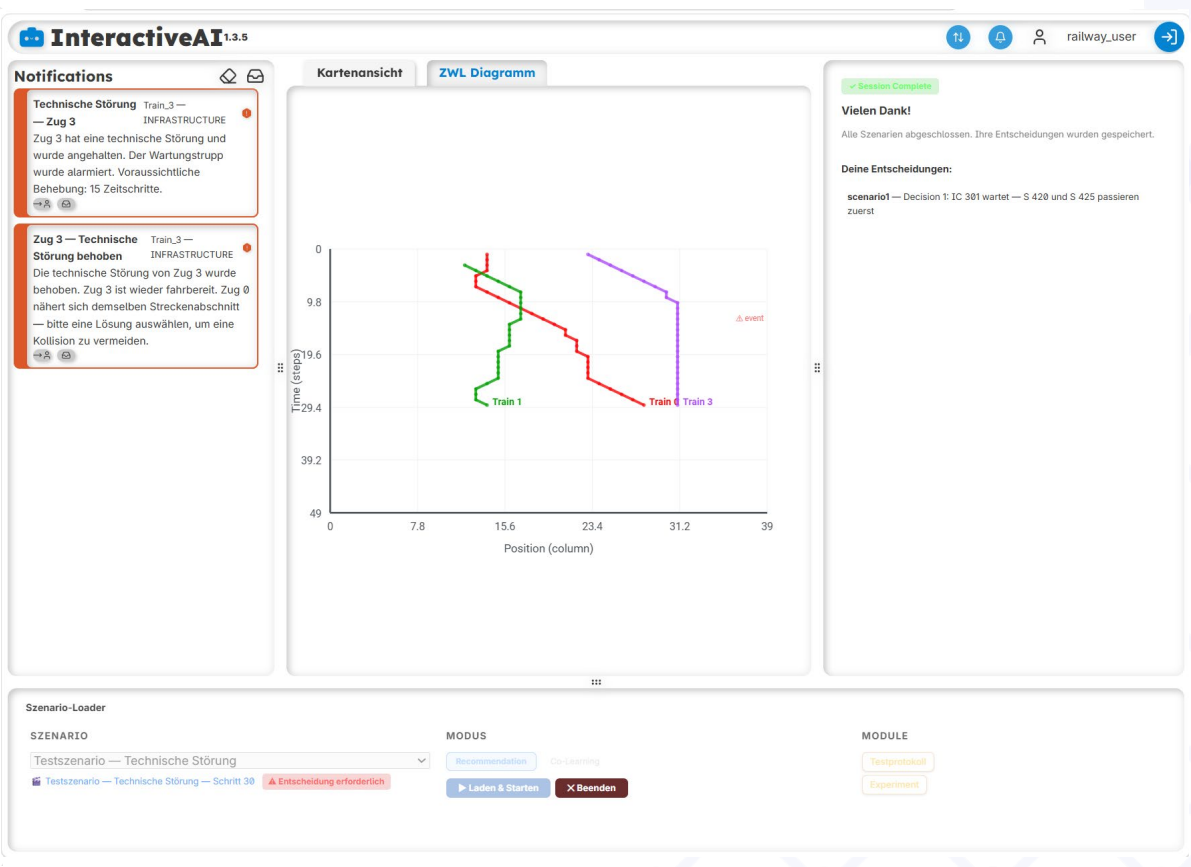


Co-learning HMI

Train-Dispatcher HMI showing a technical train collision event with a notification interface and a human feedback window.



Flatland-Train-Map view



Time-Position-Map view (ZWL/Marey-Diagram)

Application domain

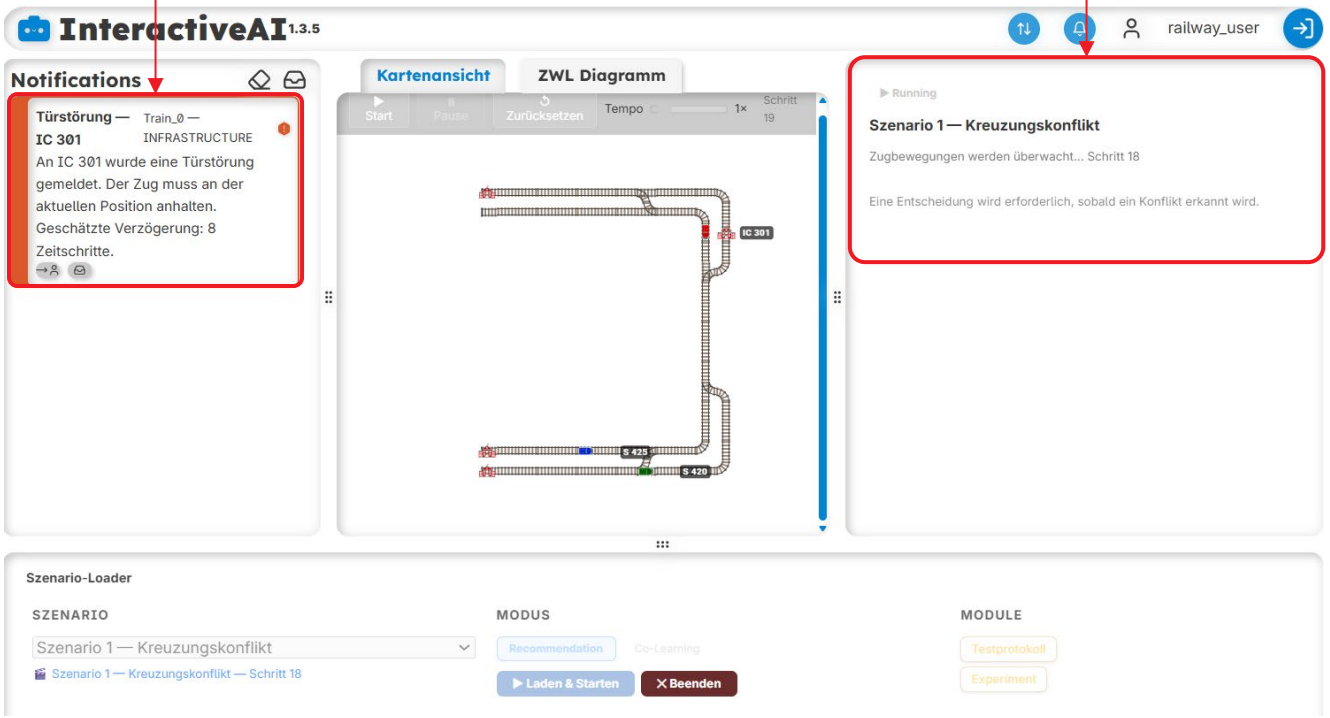
AI4 REALNET

ai4realnet.eu

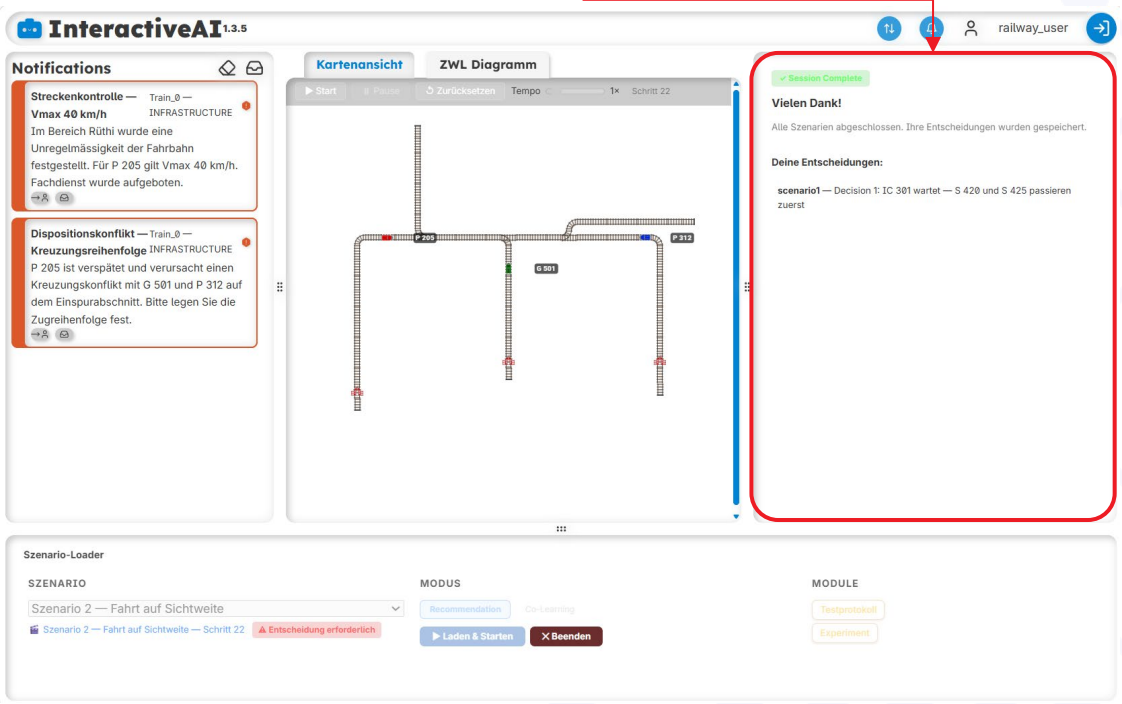
FHNW

Co-learning HMI

Train-Dispatcher HMI showing a train door malfunction with an additional information and interaction window



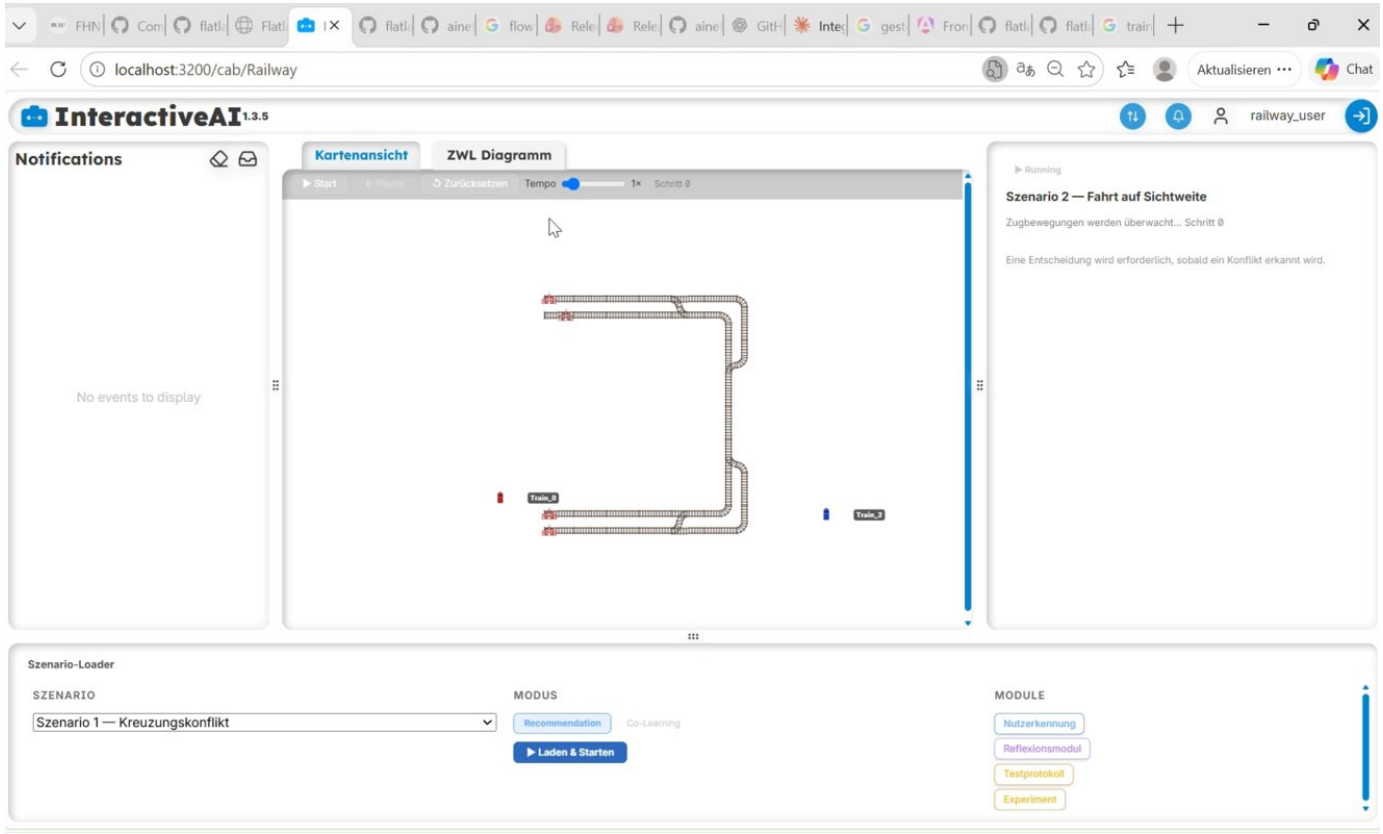
Train Door Malfunction (Failure type INFRASTRUCTURE)



Disposition Conflict (Failure type INFRASTRUCTURE)

Application domain

Co-learning HMI - Video demonstration

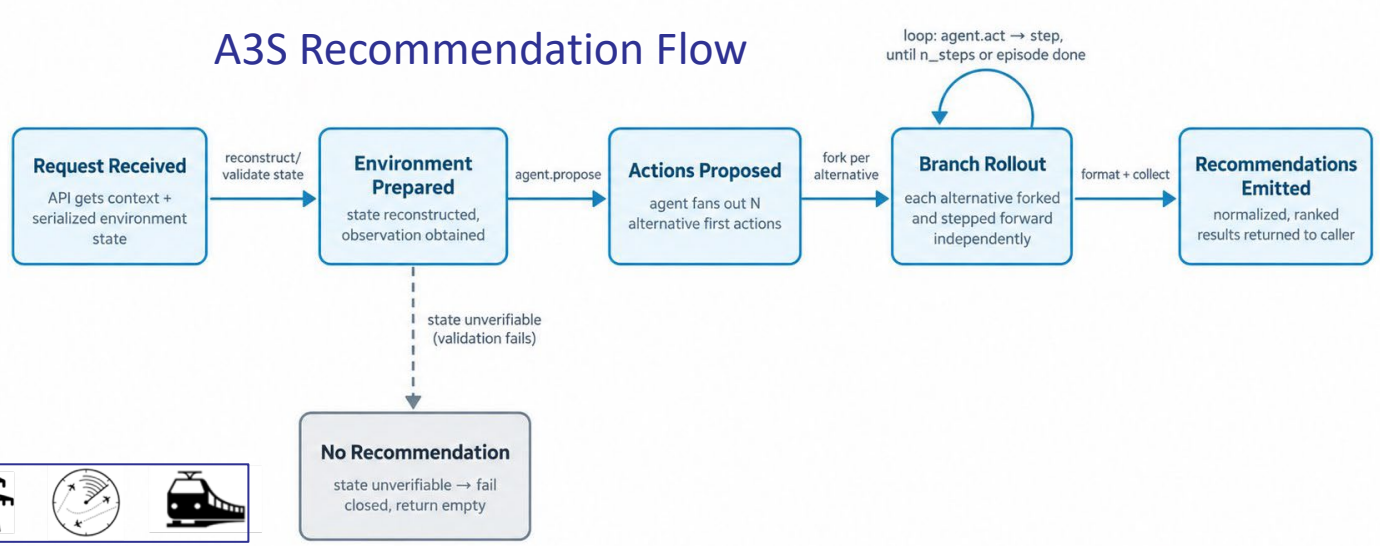


Scan the QR code for the video

Application domain

Agent as a Service

- Supports operators in action selection processes by providing them with more information about the recommended actions
- Projects KPIs of proposed actions several states in the future
  - KPI projection helps the operator decide if the action proposed by the RL agent will have any unintended consequences down the line (for future *n* steps)
  - Ensures that simulator/deployment and A3S service have fully synchronized state for realistic KPI values
    - Either through env serialization or action / history replay

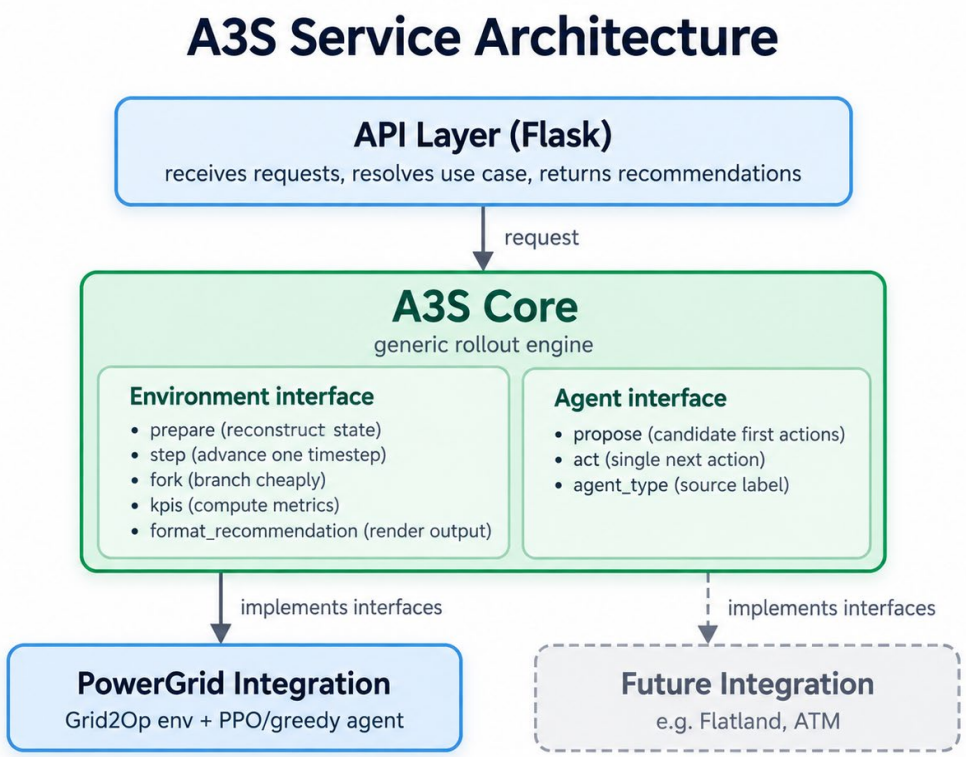


Application domain

ENLITEAI

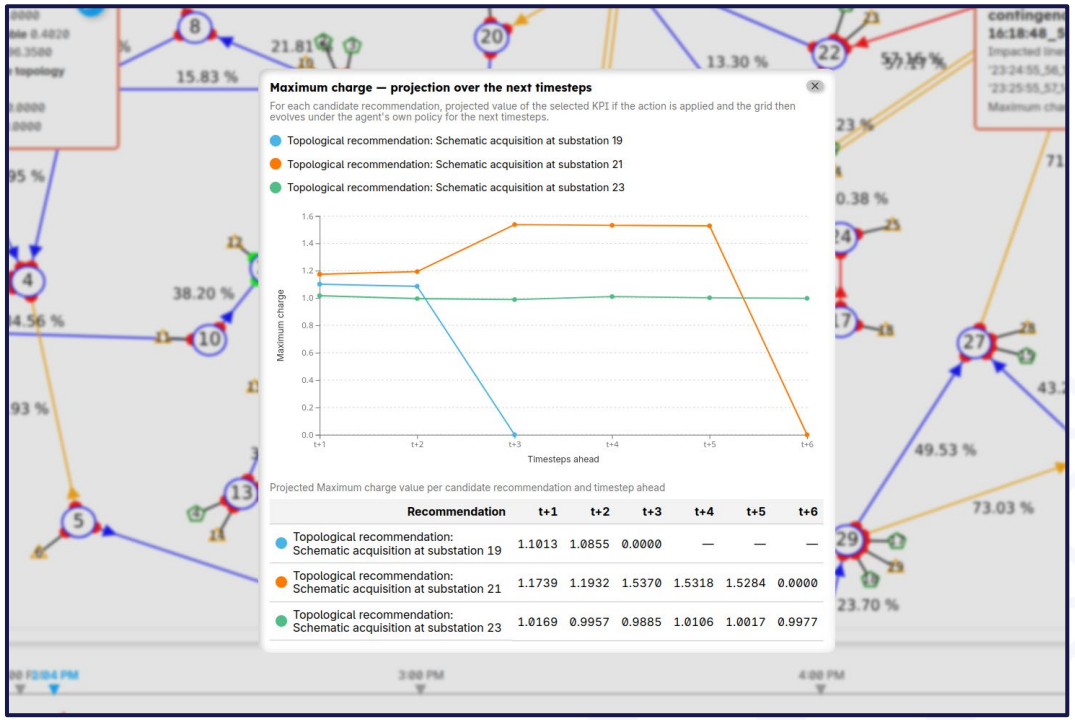
Agent as a Service

- Agent and Environment agnostic
  - Works with existing environments and RL agents that implements minimal A3S interface
  - To add a new agent or a support for new environment, user only needs to implement the Agent and / or Environment Interface
- Implemented as separate standalone Docker action recommendation service for InteractiveAI

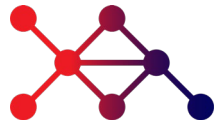


Agent as a Service

- Example KPI projection
  - All recommendation actions have similar initial KPI values, but their futures show that 2/3 actions lead to the failure down the line
  - Potential cause of this phenomenon: RL agent is too "greedy" - it optimizes mostly for the current step and does not consider the future
  - A3S separates similarly performing actions at the step zero (KPI projection based on the current state) from consequence of those actions when projected in the future



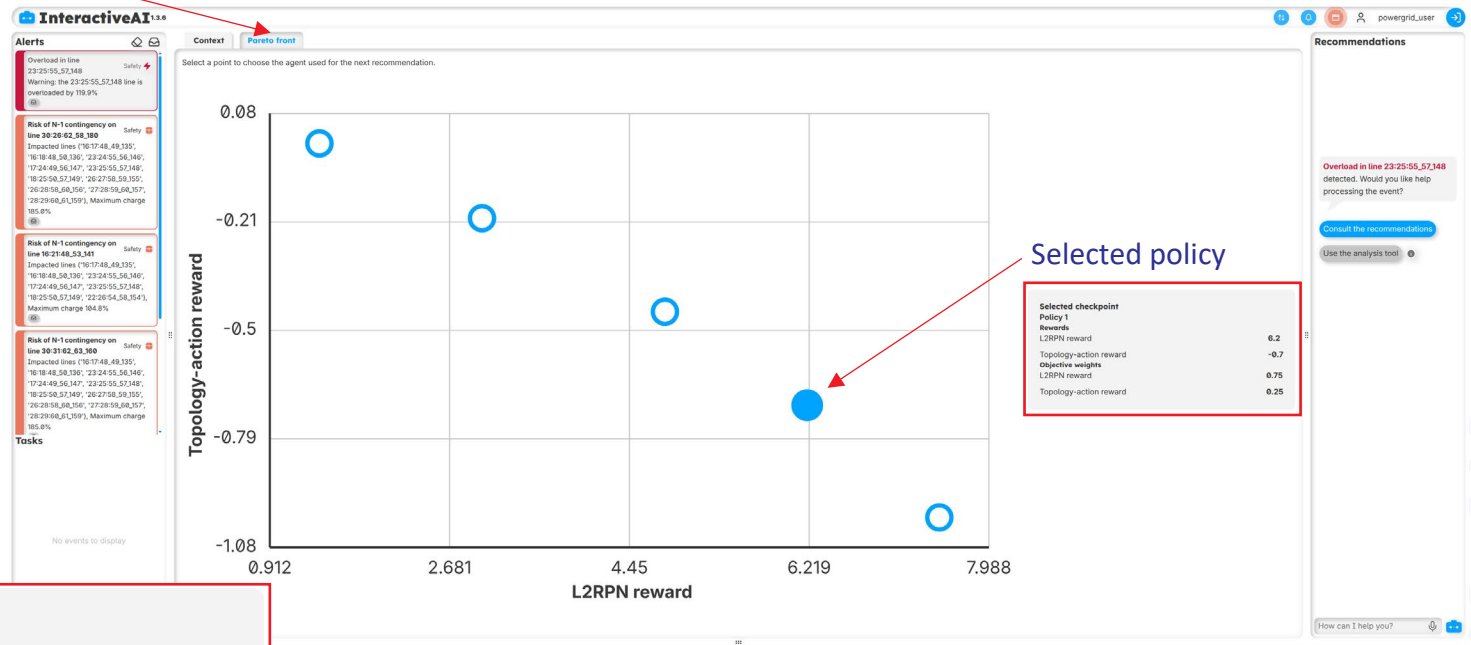
Multi-Objective Reinforcement Learning (MORL)



- MORL extends Reinforcement Learning to multiple objectives instead of optimizing a single reward.
- It is useful when objectives conflict or require trade-offs, such as efficiency vs safety.
- MORL learns policies that represent different trade-offs between objectives, ideally approximating a Pareto-optimal set of solutions.
- This provides flexibility at decision time, the preferred trade-off can be changed without necessarily retraining the agent.
- This capability has been integrated into the InteractiveAI platform allowing different policies to be selected during decision process.
- Users can select the policy that best matches the current situation or operational priorities.

Multi-Objective Reinforcement Learning (MORL)

Before consulting for recommendation user can select different policy

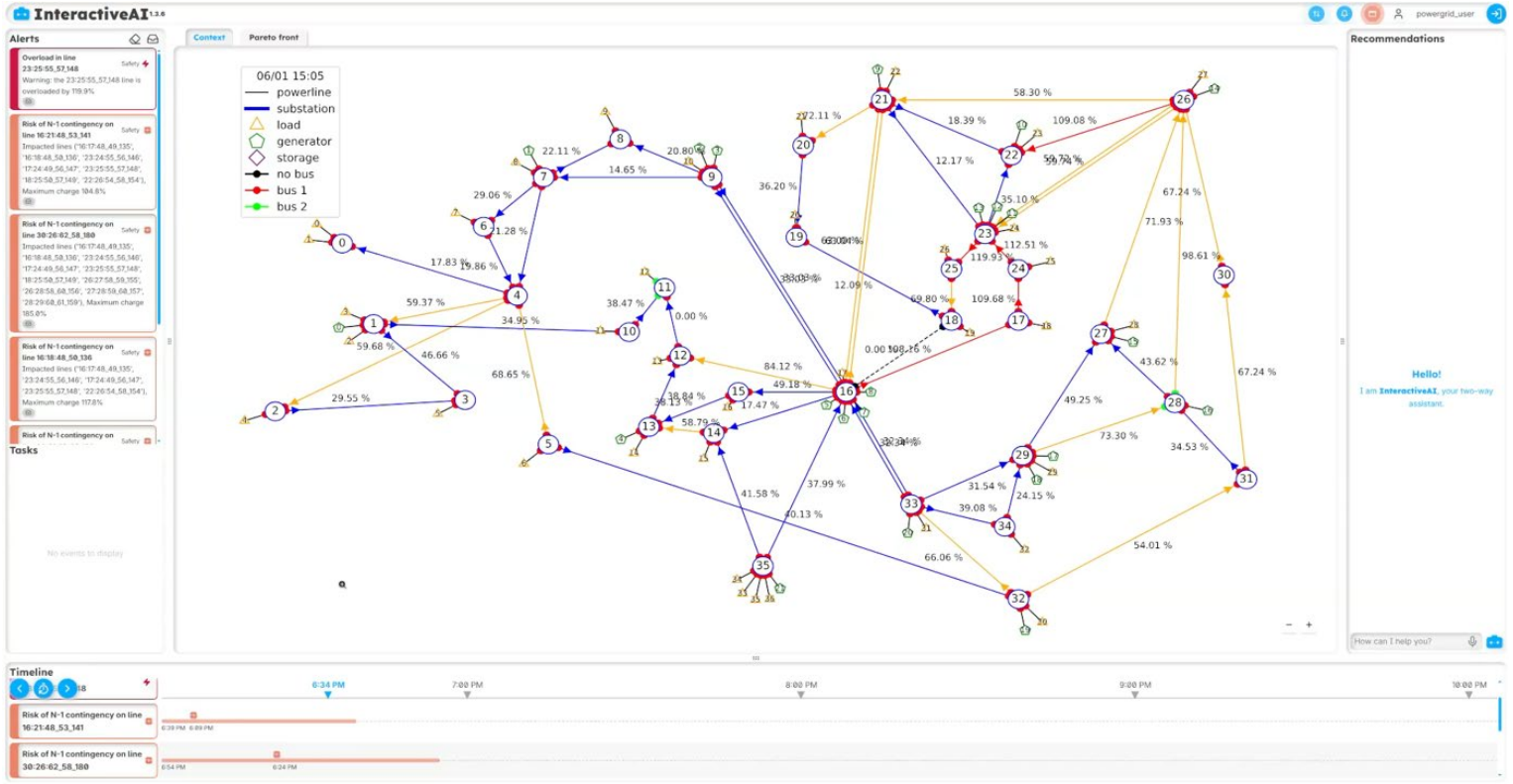


Agent's performance

Objective weights during training

Selected checkpoint
Policy 1
Rewards
L2RPN reward **6.2**
Topology-action reward **-0.7**
Objective weights
L2RPN reward **0.75**
Topology-action reward **0.25**

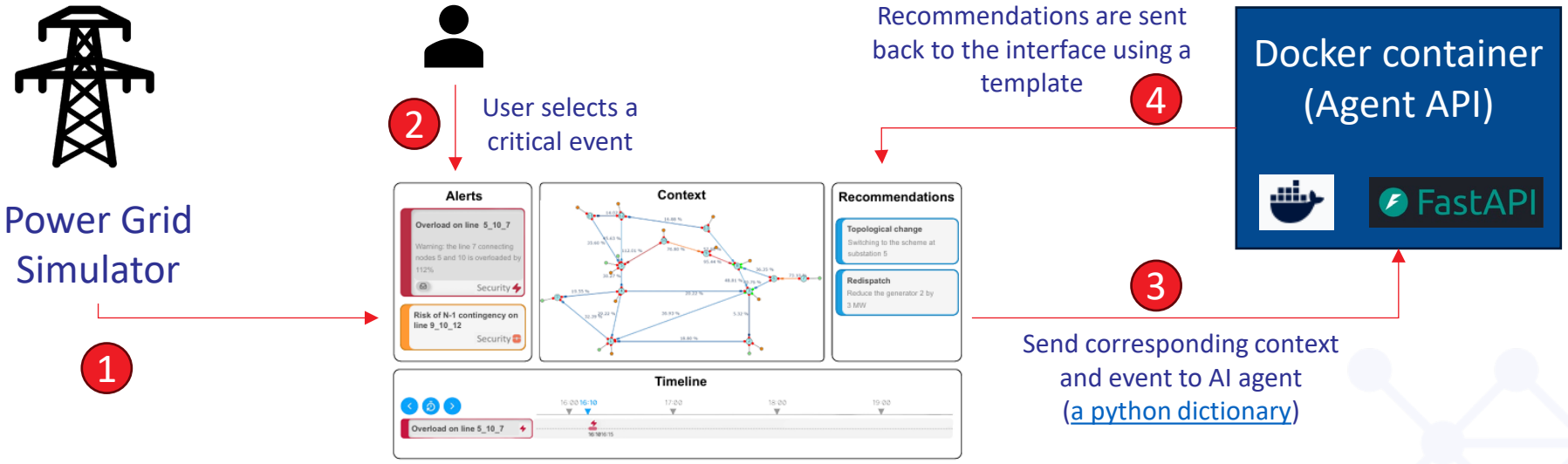
MORL Demonstration



Scan the QR code for the video

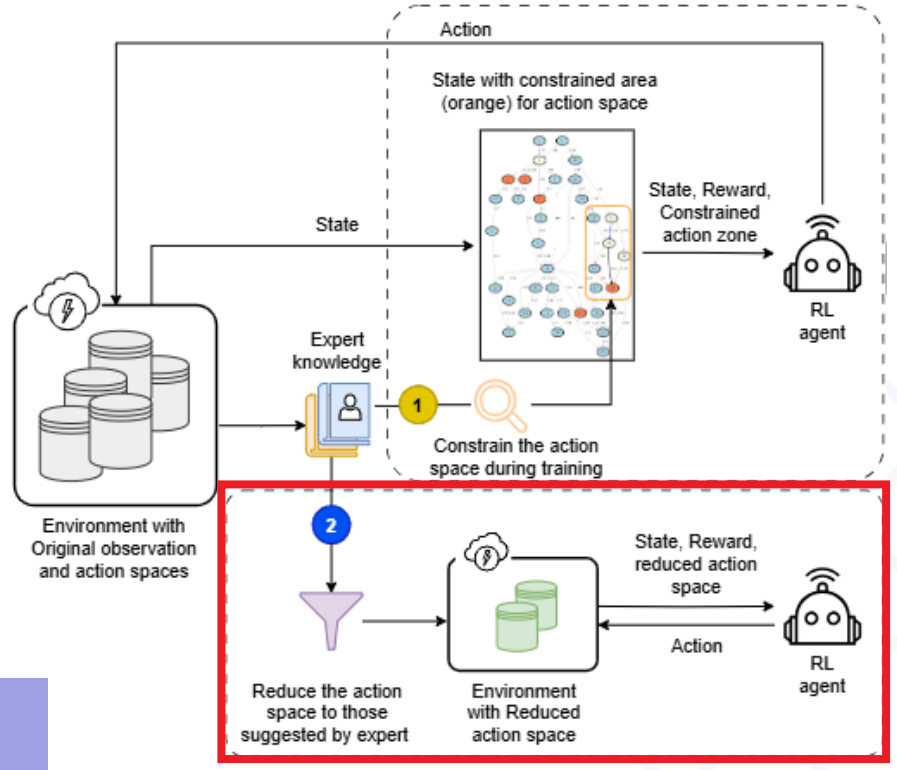
General AI agent Integration scheme

- The FastAPI python library could be used for the development of the agent API
- A docker ( ) container should be used for deployment of the AI agent
  - All the dependencies should be installed in the docker
  - Docker should expose the API interface



Expert Agent

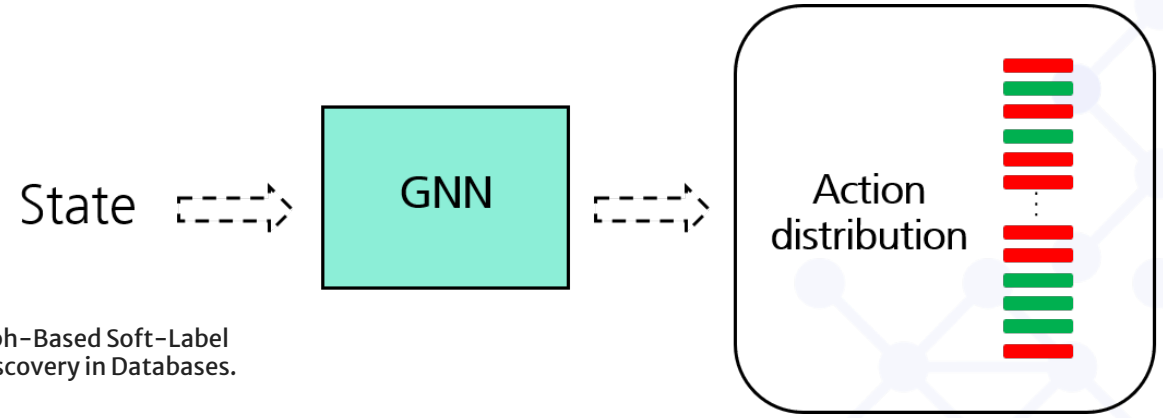
- **Main idea** = Exploit the Expert Knowledge to reduce the action space to most relevant ones and improve the scalability or RL agents
    - Heuristic-based
    - RL-based
- A set of well-known heuristics are combined with RL to select the best topological & generation redispatch actions



Submitted as a part of D2.4 in WP4 and integrated in last release of AI4REALNET framework D1.4

SoftGNN - Overview

- **Soft-label Imitation Learning**: captures multiple viable actions
- **Graph Neural Networks (GNNs)**: encode grid topology & spatial dependencies
- **Agent framework (SoftGNN)**: integrates topology reversion, N-1 security evaluation
- Significant performance gains over expert & state-of-the-art DRL for topology control

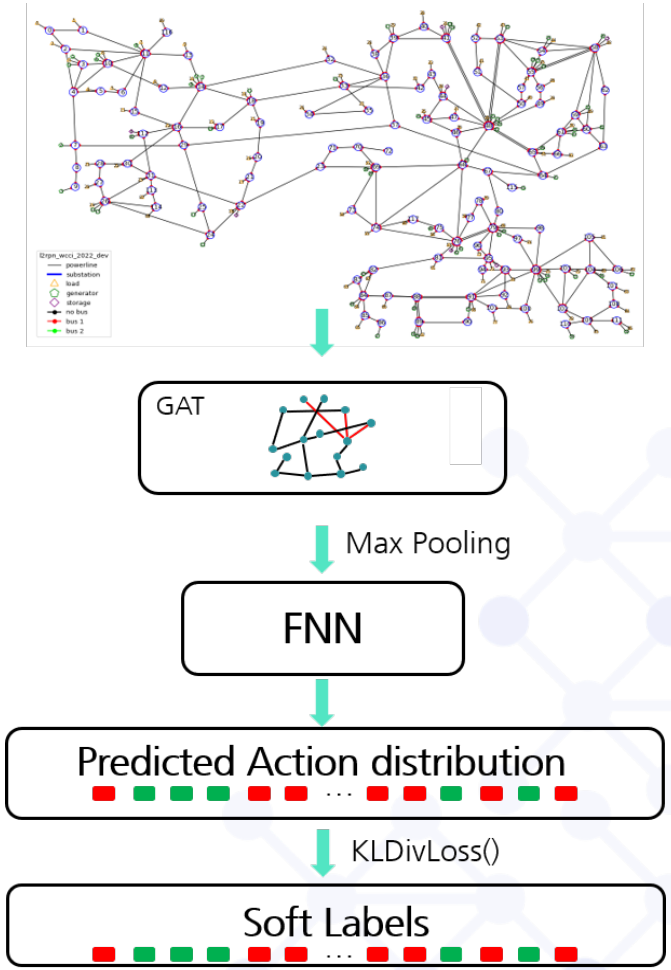


Hassouna, M. *et al.* (2026). Learning Topology Actions for Power Grid Control: A Graph-Based Soft-Label Imitation Learning Approach. In: Dutra, I., *et al.* Machine Learning and Knowledge Discovery in Databases. Applied Data Science Track and Demo Track. ECML PKDD 2025

SoftGNN - Architecture

- **Architecture**: Graph → GAT layers → pooled embedding (max) → action distribution ↔ KLDivLoss()

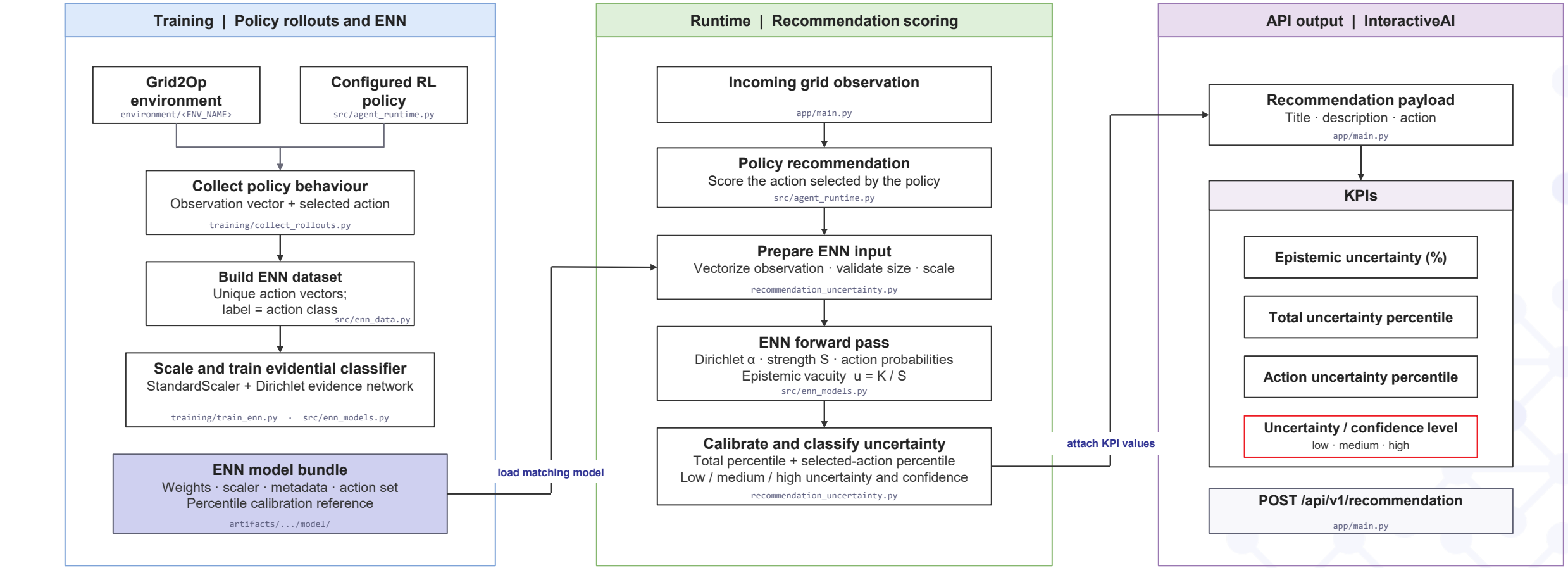
- **SoftGNN Agent**
  - Activates if max line loading ρ_max≥90%
  - GNN predicts action distribution → rank actions
  - Validate with simulation → pick first feasible action reducing overload



SoftGNN - InteractiveAI Integration

- Packaged as a Docker container
- Input: the event plus the Grid2Op observation from InteractiveAI's context;
- Output: one recommendation with title, description, the Grid2Op action, and the KPIs type_of_the_reco and efficiency_of_the_reco
- Flow: PowerGrid simulator detects an overload and pauses → operator requests recommendations in InteractiveAI → cab_recommendation calls the agent → the operator applies the action → the simulator resumes and applies it

Uncertainty Quantification



Real-time processing

Pre-saved artifact

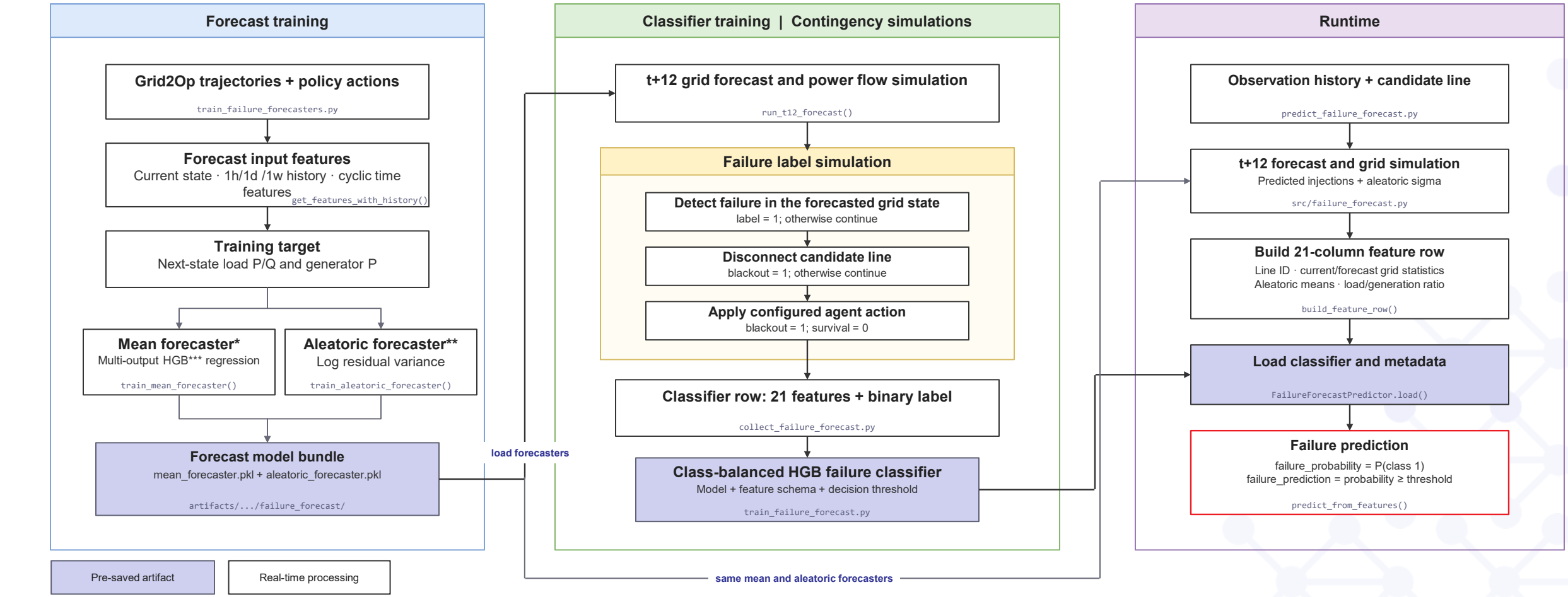
Generic module that could be used for all the agents

Application domain

ai4realnet.eu

INESC TEC

AI Agent Failure Prediction



***Mean forecaster:** predicts the expected future grid injections: load active/reactive power and generator active power. Uses **multi-output HGB regression**, tuned with **Optuna**.
****Aleatoric forecaster:** estimates the data-driven uncertainty around those predictions. Uses **multi-output HGB regression** trained on the mean model's **log-transformed squared residuals**, converted into standard deviations (σ).
***** **HGB regression:** Histogram Gradient Boosting regression

Application domain

AI4 REALNET | ai4realnet.eu

INESC TEC

AI4REALNET

AI4REALNET has received funding from European Union's Horizon Europe Research and Innovation programme under the Grant Agreement No 101119527 and from the Swiss State Secretariat for Education, Research and Innovation

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.