



**Cyber-physical systems
Industrial applications in the CPSwarm Project**



Alessandra Bagnato (Softeam R&D)

Softeam R&D (alessandra.bagnato@softeam.fr)



Overview

- The CPS world: Concept and applications
- Background
- CPSwarm Vision
- CPSwarm CPS Application scenarios
 - Case studies on CPS in Automotive: Managed Transportation
 - Case studies on CPS in Logistic: Swarm Logistics
 - Case studies on swarm drones: Heterogeneous swarms of ground rovers and UAVs
- CPSwarm Project proposed advance



CPS Summer School 2017

Designing Cyber-Physical Systems – From concepts to implementation



The CPS world

- According to the NIST definition,
 - *“Cyber-physical systems (CPS) are engineered systems that are built from, and depend upon, the seamless integration of computational algorithms and physical components”.*



CPS principal mission involves interaction with the physical world, like e.g., in cars, medical devices, scientific instruments, etc. The dynamics of such systems can evolve very fast and are dependent on variations from the surrounding environment or from other interacting systems

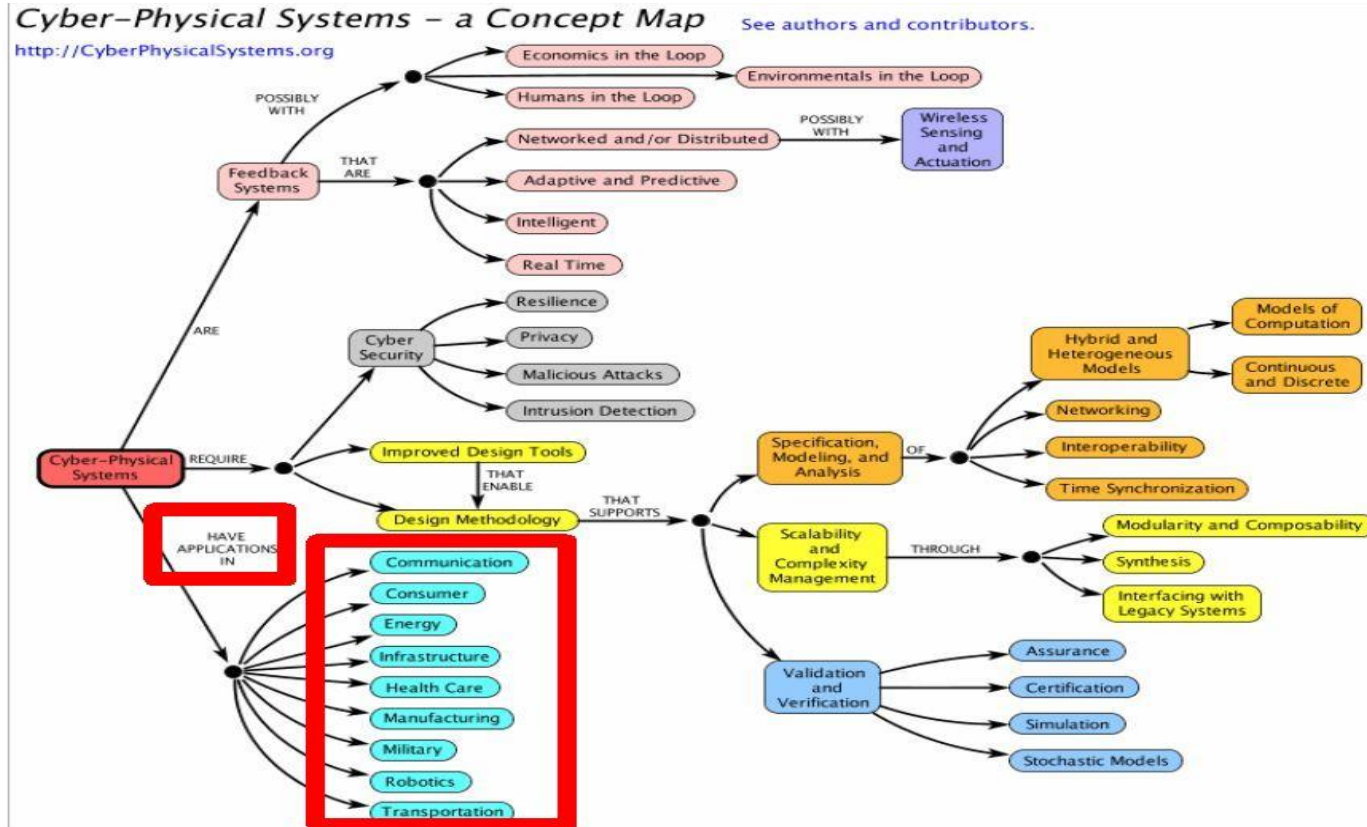
Background

- Computers are embedded in everyday objects, mostly invisible to us. They are able to perform computations on their own with their embedded intelligence.
 - Computers become smaller, more capable, ubiquitous
 - ~6.8 billion mobile phone accounts
 - 80 processors & >100M LOC in a high-end vehicle
- Furthermore, these computers are networked and link the real world to the virtual one with sensors and actuators. Such entities are called **Cyber Physical Systems (CPS)** – strongly interconnected hardware and software components, being applied in the Internet of things, smart grids, smart mobility- and smart factory concepts.



Cyber-physical systems will be able to exchange data with each other, access web services, and interact with people: new challenges are including increased dynamics, connectivity, and complexity calling for features like adaptability, scalability, robustness, self- configuration, self-healing, etc.

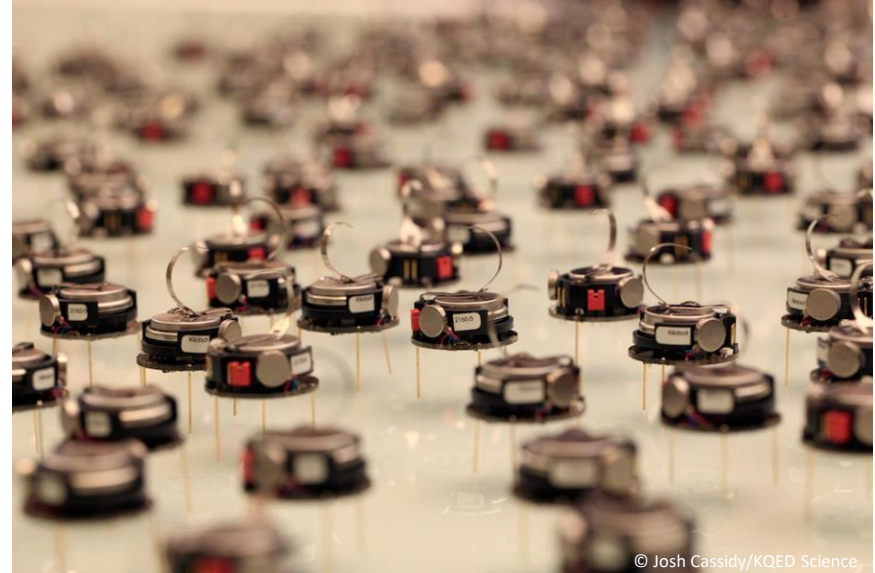
CPS Applications



CPSwarm Vision

Interactions amongst CPS might lead to new behaviors and emerging properties, often with unpredictable results.

These interactions can become an advantage if explicitly managed since early design stages.



© Josh Cassidy/KQED Science

High-Level Objective

CPSwarm proposes a new science of system integration and tools to support engineering of CPS swarms.

*CPSwarm tools will ease development and integration of **complex herds** of **heterogeneous CPS** that collaborate based on local policies and that exhibit a **collective behavior** capable of solving complex, industrial-driven, real-world problems.*





CPSwarm at a Glance

- CPSwarm is a **36-months Research and Innovation Action (RIA)** funded under H2020 call ICT-01-2016
- **Scope: science of system integration** in the domain of **swarms of CPS**
- **8 partners** (3 Research Institutes, 1 University, 2 Large Enterprises, 3 SMEs) from **6 EU countries**
- Around 4.9 M€ total costs (578 PMs)



The CPSwarm Consortium

Coordinator



IT



DE



ES



AT



AT



AT



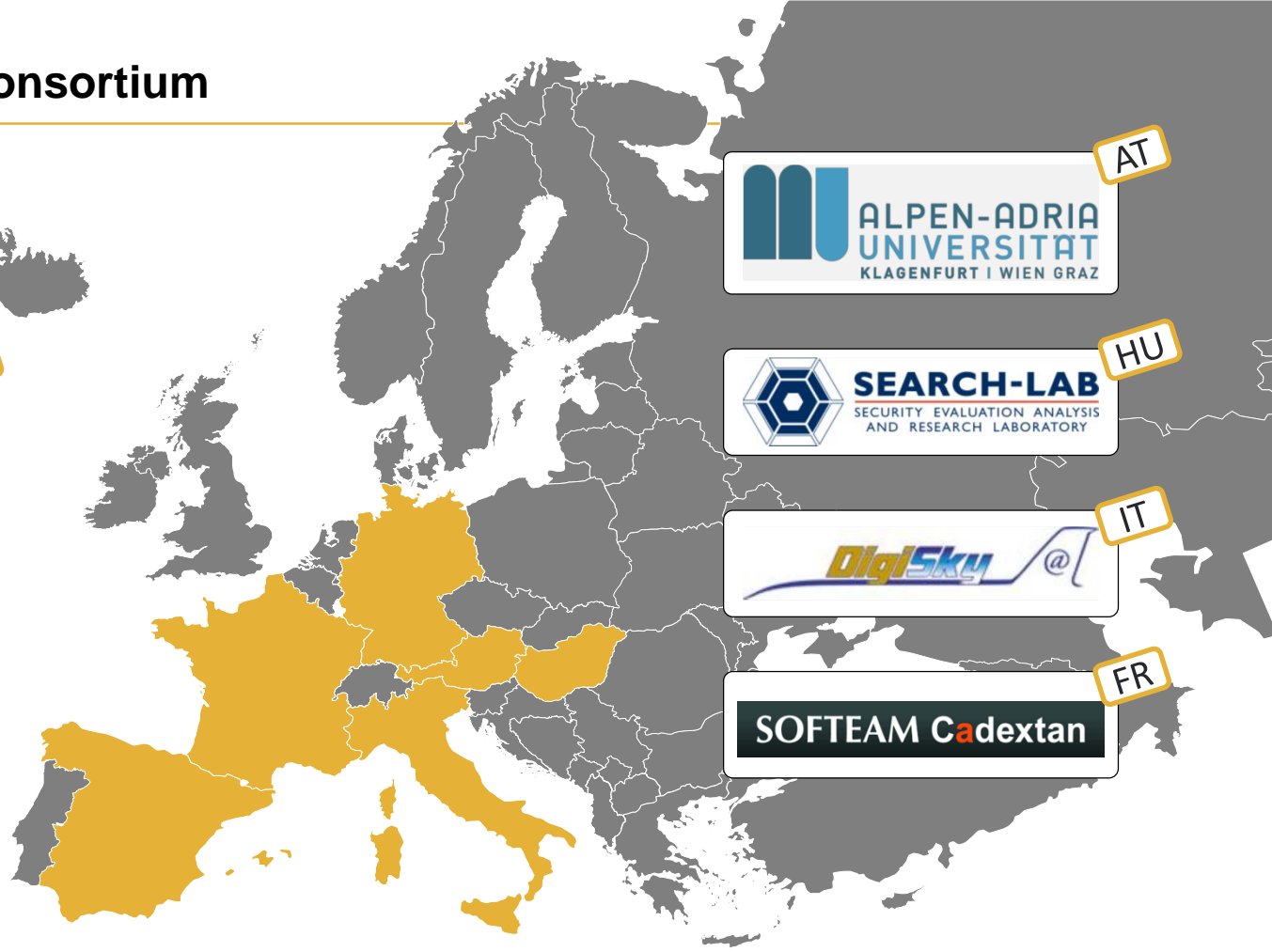
HU



IT



FR



Application Scenarios

Three reference Application Scenarios drive the collection of requirements for the development of the **complete CPSwarm toolchain** supporting the *engineering and deployment of CPS swarms*



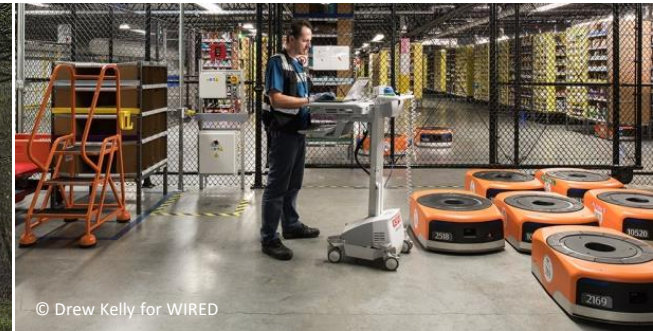
Phantom_Glacier: Image courtesy DJI

Swarm Drones



© dailymail.co.uk

Automotive CPS



© Drew Kelly for WIRED

Swarm Logistics Assistant

CPSwarm application scenarios contexts

- Cpswarm offers the integration of systems and tools for supporting the CyberPhysical Swarms
- TTTech, Digisky, Robotnik will provide the tools and platforms for the experimentation in automotive, swarm drones and logistics scenarios



Automotive CPS: TTECH – Automotive Cyber Physical Systems Use Case

- **Objectives:** Enhancing and carrying on the developments for the (near) autonomous platform for autonomous operation.
- Applications for **collective driving** with a focus on **autonomous driving vehicles intended for freight transportation**
 - independent vehicles could join or leave a swarm at any point during the journey
- Laboratory level demonstrator (TRL 3 to TRL 4, demonstration in breadboard lab environment)
 - E.g., trucks, vans or cars and connecting them via kind of an electronic drawbar.

Automotive CPS – Relevant technologies

- Software Systems operating in vehicle environment are based on **Electronic Control Units** (ECUs) supporting a complex structure of real time components, acting on thousands of attributes adjusted to refine the car's character, fulfil the regulations, etc.
- The collection of requirements driving the systems design and the management of the software design process are supported by ad-hoc tools (e.g., IBM Rational Doors)
- High-level software design (structure and behaviour) benefits from general UML tools (e.g., IBM Rational Rhapsody or Modelio)
- **AUTOSAR** (AUTomotive Open System ARchitecture) is the relevant standard
 - specific tools (e.g., Vector's PREEvision) are used to support software development, model-based specification of electronic vehicle systems and design of vehicular network
- **Simulation** and **modelling** of software functionalities (e.g., control algorithms) are based on tools like ETAS Ascet or Simulink/MATLAB. These tools are also used for the generation of real-time, production code.

Swarm Logistics Assistant: Robotnik

Focus on robots and rovers designed to **assist humans in logistics domain**

- Scan the entire area of the warehouse and share the acquired information
- Collect information about the maps of the entire area
- Collect additional information implicitly e.g. room temperature, presence of humans, detection of in-path obstacles etc.
- Join forces to move a heavy obstacle from one place to another

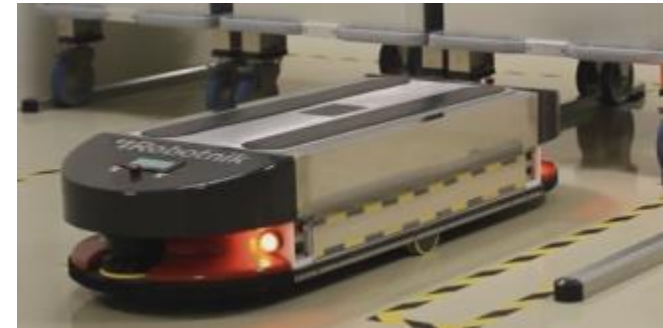


Swarm Logistics Assistant – Relevant technologies

- The adopted Operating System for ground robots is **Robotic Operating System** (ROS) - a de-facto standard for robotics
 - **Modular** architecture enabling the development of custom packages and the integration of third party tools
 - **A complete toolchain** facilitates interaction, control and monitoring of robots also through GUIs (e.g., Rviz and RQT) consisting of a 3D visualizer and showing how robots perceive, measure and interact with the environment
- **Robot description** is supported by **URDF**, defining two types of components
 - **Links** – fixed parts of a robot including 3D models (enabling computation of possible collisions and feeding 3D visual simulators)
 - **Joints** – represent how links are connectedand a hierarchy-based modelling to describe any kind of robot
- **Robots simulation** is enabled by ROS and Gazebo



Turtlebot 2



AGVS

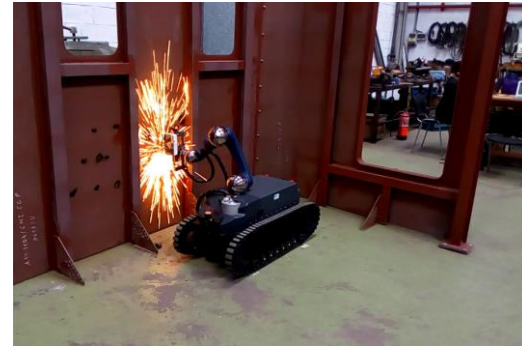
CPS Use Cases: the Robotnik context

- Robotnik is a Service Robotics Manufacturer: mobile bases and manipulators:
 - **Unmanned Ground Vehicle (UGV)**'s - An **unmanned ground vehicle (UGV)** is a [vehicle](#) that operates while in contact with the ground and without an onboard human presence
 - Mobile manipulators
 - Omnidirectional mobile platforms
 - ROS based robots



CPS Use Cases: the Robotnik context

- Custom application providers
 - Surveillance and defense
 - Agriculture
 - Maintenance and monitoring



CPS Use Cases: the Robotnik context

- Service Robotics: Logistics
 - Focusing in logistic services.
 - Optimization of productive processes.
 - Collaborative robots



CPS Use Cases: the Robotnik context

- Intra-hospital logistics with *Automatic Guided Vehicle Systems (AGVS)*
- AGVS Robotnik is an autonomous robot created to perform transport tasks in hospitals or health centers.
- The fleet of robots is supplied with the installation on the environment and fleet management software.



Case of study: Robotnik AGVS

- <https://www.youtube.com/watch?v=Oc8AJIbJkmc>



Case of study: Robotnik AGVS

- **General vision of swarming architecture:**

- Swarming strategies examples with mobile robots
- Consensus to avoiding collisions (1 vs 1 & N vs N)
- Robot formations
- Self-organization to carry out some mission:
- Help request
- Delegate to another
- Decision-making

- **Requirements of swarming architecture**

- Sensor requirements
- Communication P2P
- Detection of another robot
- Identification
- Consensus strategies
- Interpretation of the environment
- Partial or complete knowledge of state of other robots
-



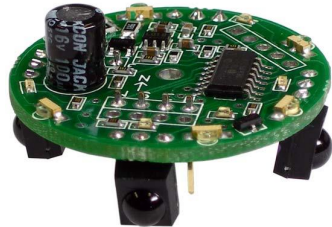
Pololu IR transceiver:

- The Pololu IR (infrared) beacon is a compact board that is used in pairs to allow robots to locate each other. Each board has infrared emitters that shine in all directions and four IR receivers for detecting the other beacon. The IR beacons have a range of about fifteen feet indoors.

Possible robots and sensors



Turtlebot II



RP-Lidar A2

<https://www.youtube.com/watch?v=9OC3J53RUsk>

TurtleBot3
BURGER



TurtleBot3
WAFFLE



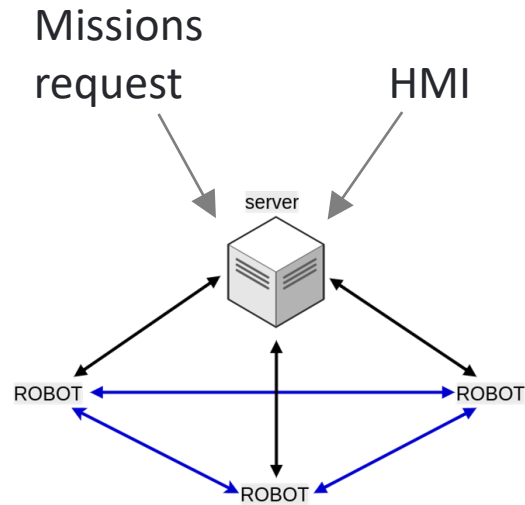
GAZEBO

Example Video

<https://www.youtube.com/watch?v=9OC3J53RUsk>

Simple logistic CPSwarm scenario

- Load/Store material



Simple scenario: what do we know

- Static positions of charge stations and load/store locations.
- State of any robot at any moment.
- List of missions required.
- Minimum battery level required for each mission.
- How to assign missions to the robots? First approximation => Assignment of the free robot closest to the first location of the mission, which also fulfills the requirement of the battery level needed.



Adding complexity to logistics scenarios:

- Adding complexity to these scenarios:
 - Heterogeneous sensor mix
 - Heterogeneous robot qualities: load capacity, omni directional movement
 - Heterogeneous device capabilities: manipulator vs load space
 - CyberPhysical Swarm with camera recognition etc
 - Automatic requests to transport something.



DGSKY – Heterogeneous swarms of ground rovers and Unmanned Aerial Vehicles (UAVs) Use Case

Detailed Use case:
Heterogeneous
swarms of ground
rovers and UAVs

SYSTEM OVERVIEW



Heterogeneous swarms of ground rovers and UAVs

- In CPSwarm we consider heterogeneous swarms of ground robots/rovers and UAVs to conduct certain missions in the surveillance of critical infrastructure like industrial or power plants, search and rescue (SAR) and fire detection.
- The use case exploits different types of robots to approach an area and support rescuers/guardians by: mapping the actual state with a 3D map, finding people, tracking them and checking on their condition using computer vision, identifying safe passages for rescuers.
- Smaller robots (rover/UAV) in the swarm are employed to map the environment and possibly move inside narrow passages whereas "bigger" robots could either guide rescuers to people.



HETEROGENEOUS SWARMS OF GROUND ROVERS AND UAVs

01

Intrusion detection
(detection of unauthorized
persons entering of the
plant area).

Follow and observe actions
of unauthorized persons in
the plant.

SURVEILLANCE

02

Generating a situation
overview of the disaster
scene in case of an
industrial plant accident
including real-time images
(vis, IR), toxic and explosive
gas leakage detection.

Finding of human casualties
or persons trapped in the
disaster area.

SEARCH & RESCUE

03

Generating a situation
overview of the disaster
scene.

Aerial or ground unmanned
systems will ensure safer
and more efficient
intervention.

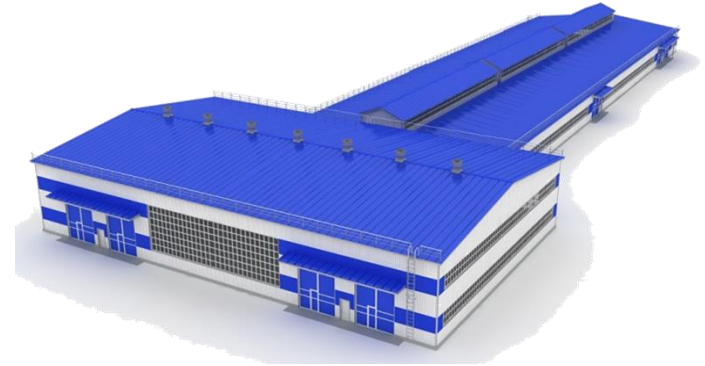
FIRE FIGHTERS

USE CASE: SYSTEM ARCHITECTURE



Operations Center

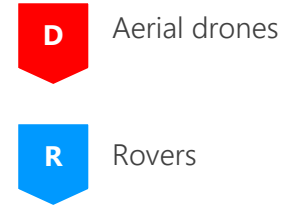
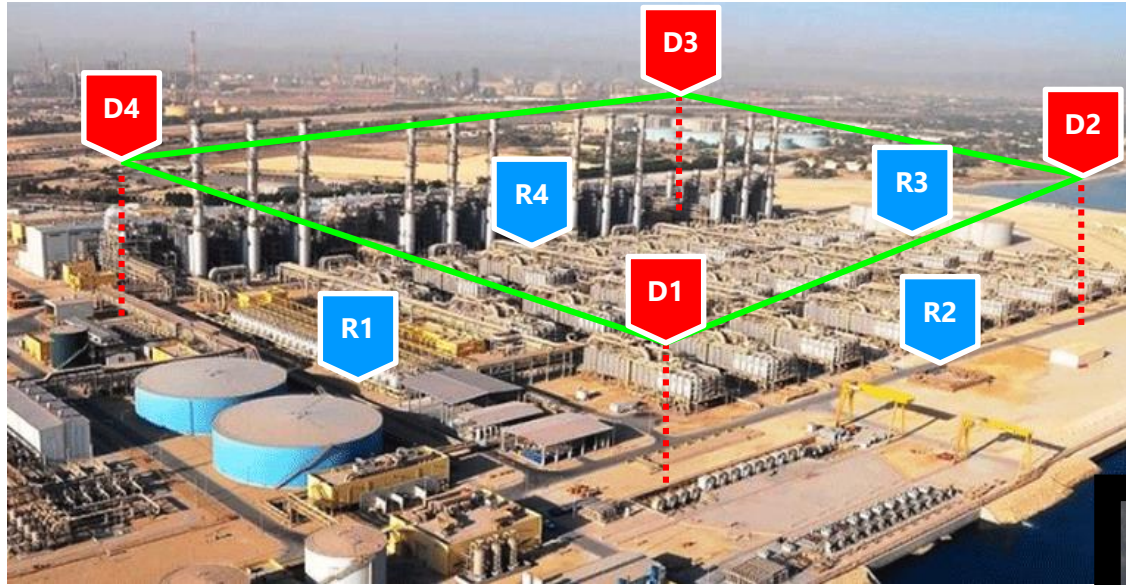
- Real-time video acquisition
- Log of all operations (GPS track, time and duration, images etc.)
- Real-time image processing (Computer Vision)
- Camera remote control from Operations Center (tilt and zoom)
- Remote control of dynamic waypoints



Patrol – Intervention

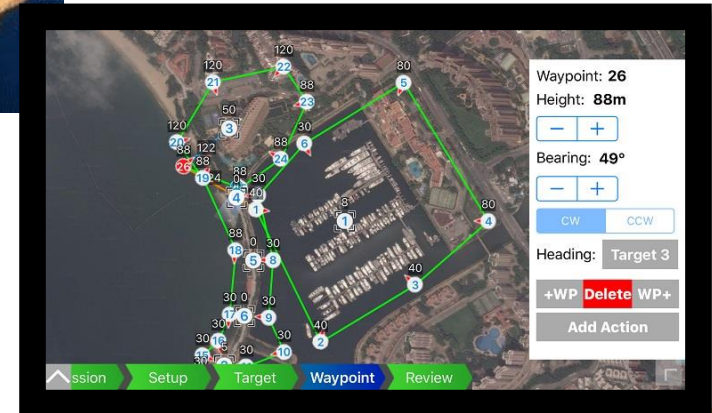
- Real-time video acquisition
- Autonomous flight
- Interaction with Operations Center by radio
- Wearable device for mission control (start, pause, stop mission, telemetry)

USE CASE: OPERATING SCENARIO



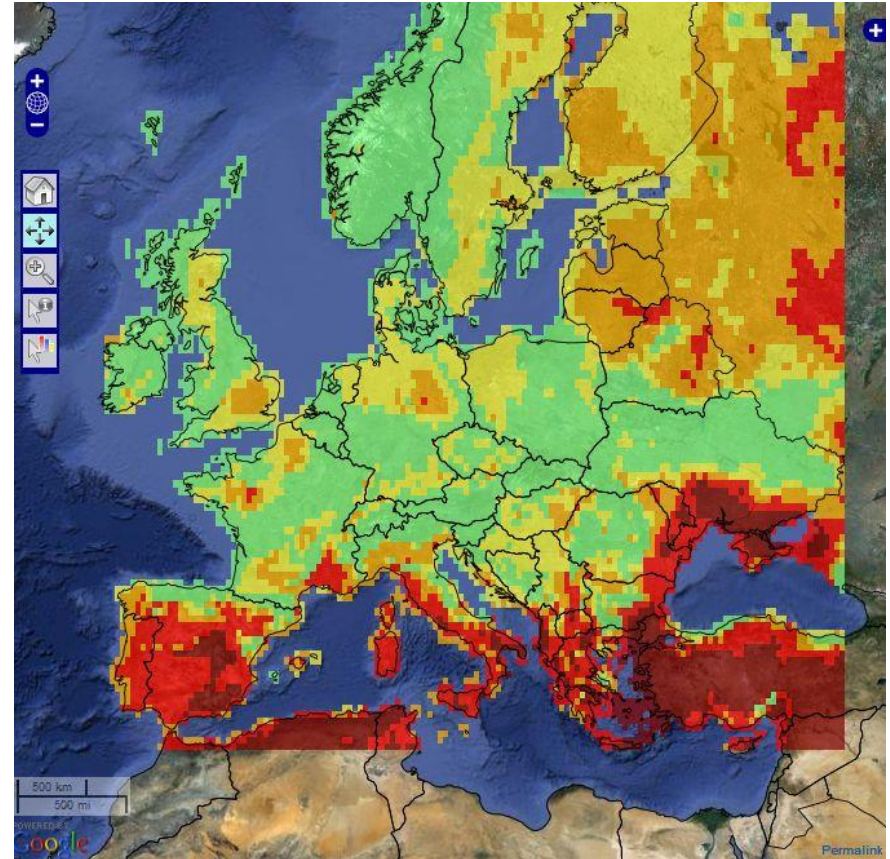
A single mission planner to define the area of intervention setting the limits, the path and the observation points.

Drones and rover are synchronized automatically, each device knows the position of others and communicates with the control center via 4G/LTE/WiFi



USE CASE: FIRE FIGHTERS

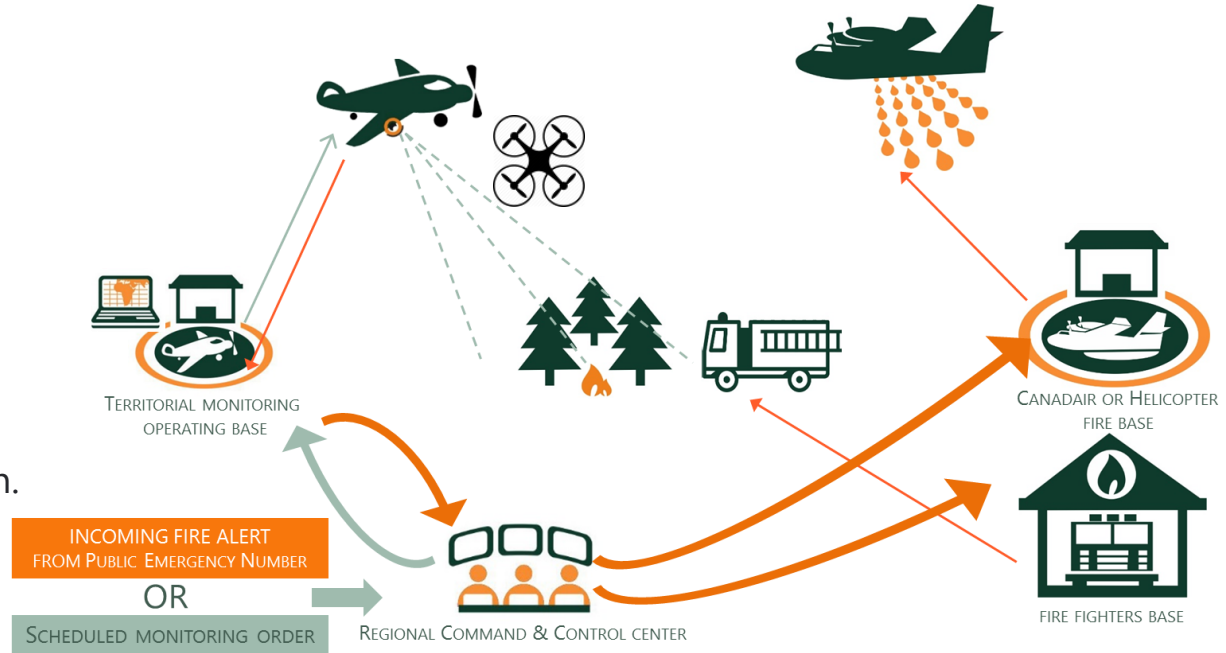
- Every year forest fires burn, on average, about 500 000 hectares in Europe.
- Each fire requires the intervention of firefighters, involving the deployment of aerial and ground assets which need to be coordinated in high risk operational areas.
- Currently, the intervention is solely coordinated by the chain of command, which may expose to high danger the firemen safety. As the fire dynamics is depending on external factors (wind vegetation, artifact), the development of mathematical models is required.
- The possibility to employ aerial or ground unmanned assets will ensure safer and more efficient intervention.



USE CASE: FIRE FIGHTERS

CPSwarm Flexible Approach

- Automatic planning of drones appropriate to the mission.
- Modelling of evolving scenarios based on complex environmental parameters.
- Coordination between manned and unmanned systems.
- Mission evolution monitored by the Command and Control Station.



ADVANTAGES OF UAS SWARM

- Drones/rovers start from the same starting point and reach their own positions autonomously.
- Strategic real-time viewpoints during inspections/SAR operations.
- A single mission for all devices: area planning, boundaries, paths and viewpoints.
- Huge reduction in the time taken for inspection/SAR in large sites/areas.
- Human error reduction: repetitiveness of the operations is carried out by an «unmanned systems».
- Increased security for supervisory staff.
- Real-time interaction with Operations Center, better coordination during the inspection/SAR operations.
- Not only visual inspection by use of different sensors: audio analysis, presence of gas, thermal cameras etc.

Swarm Drones

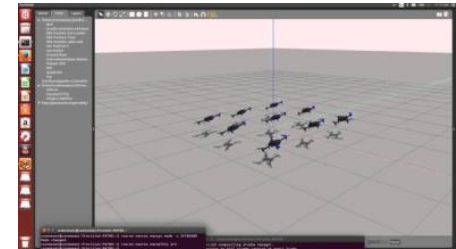
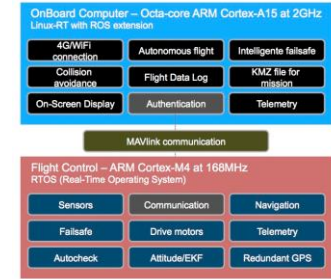
Heterogeneous swarms of ground robots/ rovers and UAVs to conduct certain missions in

- **Surveillance of critical infrastructures** like, e.g., industrial or power plants
 - intrusion detection (detection of unauthorized persons entering the plant area)
 - monitoring of actions of unauthorized persons in the plant areas
- **Search and Rescue tasks**
 - generating a situation overview of the disaster scene in case of an industrial plant accident including real-time images (VIS, IR), toxic and explosive gas leakage detection
 - finding of human casualties or persons trapped in the disaster area.



Swarm Drones – Relevant technologies

- Drones are equipped with **PX4**, a **flight control platform** capable to support the complex coordination and swarm behaviors researched
 - PX4 Flight Stack - flight control autopilot
 - MAVLink - a highly efficient, lightweight robotics communication toolkit
 - QGroundControl - a UI to configure the system and execute flights
- Simulation and modelling of software functions (e.g. control algorithms, Attitude and Heading Reference System, collision avoidance) are based on Simulink/MATLAB.
- Production-level code is tested using **HW In The Loop simulations**, (jMAVSim), or **SW In The Loop simulations** (Gazebo and ROS).
- The **model** of a drone, including HW characteristics, physical aspect and behaviour, can be created using **SDF** i.e., an XML format that describes objects and environments for robot simulation, visualization, and control.





SYSTEM ARCHITECTURE

DGSKY – Heterogeneous swarms of
ground rovers and Unmanned Aerial Vehicles
(UAVs) Use Case

UAV ARCHITECTURE

The UAV systems that will be developed for the CPSwarm project will be based on the following architecture:

- Robust carbon frame, lightweight and easily transportable
- Modular 4-8 engines system to have a high payload capacity and redundancy
- Modular gimbal to adapt to different types of payload (standard camera, night, thermal)
- Flight control with redundant IMU, GPS, compass, barometer
- Collision detection system
- Datalink for telemetry and video streaming
- 4G/WiFi connection
- Ground Control Station on desktop or tablet
- Wearable device integration



UAV ARCHITECTURE

UAV type	Quadcopter, Octocopter
Flight control	ARM Cortex M4 + FPU at 168 MHz
OnBoard Computer	Cortex-A15 2Ghz + Cortex-A7 Octa core
Operating System	Ubuntu 14.04, Linux-RT, ROS Indigo, mavROS
Size	60 x 60 cm
MTOW	3,0 Kg
Payload	Up to 800 g
Flight time incl. payload	30 min.
Range	4.000 m ASL, 1.000 m AGL
Max airspeed	15 m/s
Max climb rate	5 m/s
Datalink	2,4 GHz XBee, 488 MHz radio, 4G/LTE, WiFi
Flight modes	GPS Mode, Attitude Mode, Manual Mode
Failsafe	Automatic landing, Return-To-Home



UAV ARCHITECTURE

OnBoard Computer – Octa-core ARM Cortex-A15 at 2GHz
Linux-RT with ROS extension

4G/WiFi connection

Autonomous flight

Intelligente failsafe

Collision avoidance

Flight Data Log

KMZ file for mission

On-Screen Display

Authentication

Telemetry

MAVlink communication

Flight Control – ARM Cortex-M4 at 168MHz
RTOS (Real-Time Operating System)

Sensors

Communication

Navigation

Failsafe

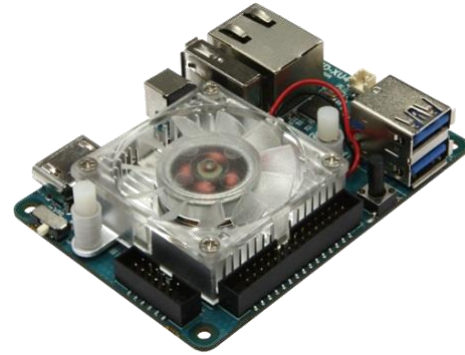
Drive motors

Telemetry

Autocheck

Attitude/EKF

Redundant GPS



FLIGHT CONTROL & AUTOPILOT: WIDE AIRFRAME SUPPORT

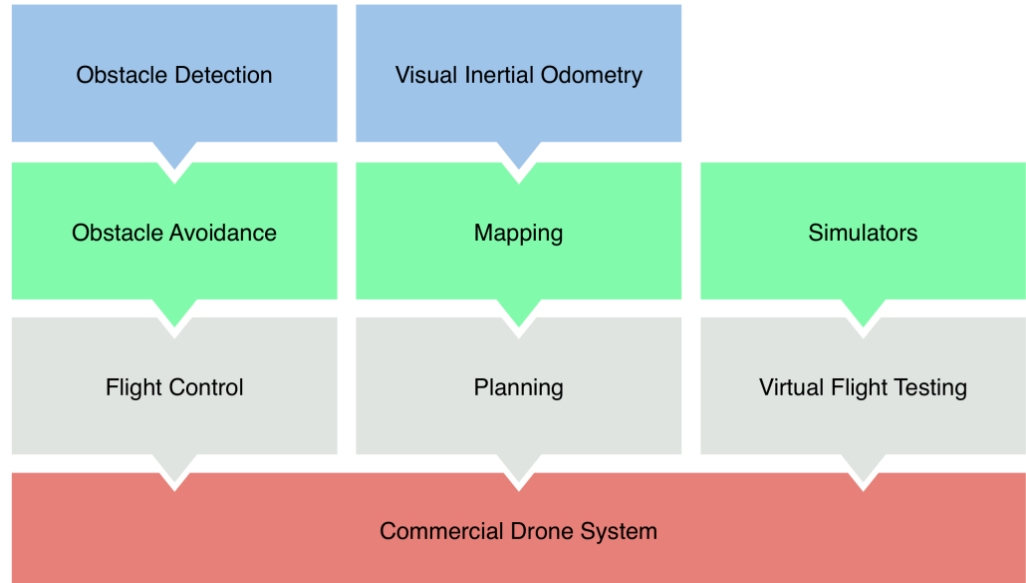
Our UAV systems are made based on PX4 flight stack and flight controller. The PX4 flight stack is a complete autopilot solution and it consists of several customizable software packages.

- **PX4 Flight Stack**: a complete flight control solution for multicopters, planes, VTOL aircraft or any ground robot.
- **PX4 Middleware**: a highly efficient, lightweight and blazing fast robotics communication toolkit.
- **QGroundControl**: modern, mobile and desktop user interface to configure the system and execute flights.



FLIGHT CONTROL & AUTOPILOT: FEATURES

- Optical Flow and Visual Inertial Odometry.
- Collision avoidance.
- Mission planning for autonomous flight.
- Simulation.
- Hardware independent.



FLIGHT CONTROL & AUTOPILOT: **FEATURES**

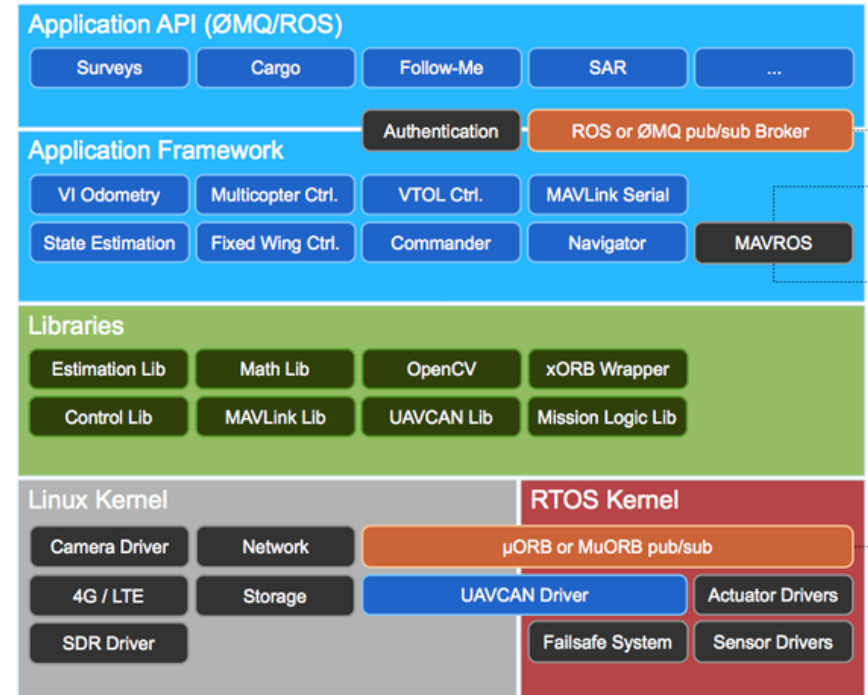
- Strong Ecosystem and Partners



FLIGHT CONTROL & AUTOPILOT: DEVELOPMENT SYSTEM

- PX4 is easy to use and easy to configure and offers a consistent user interface experience across mobile and desktop.
- Software architecture is based on Real-Time Operating System NuttX, hardware independent.
- New modules, libraries and core code can be developed in C++ language.
- ROS Robot Operating System is fully supported.

ROS

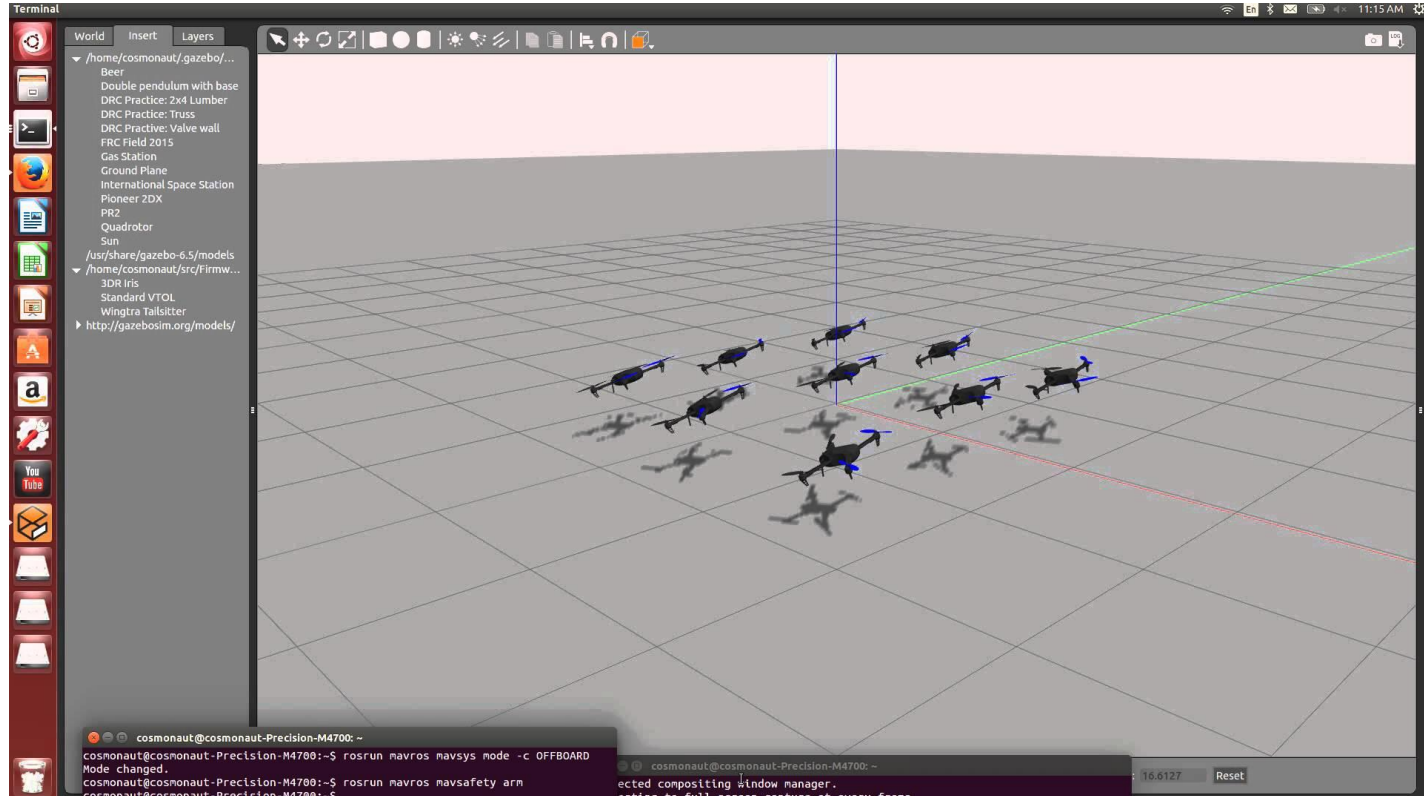


FLIGHT CONTROL & AUTOPILOT: SIMULATION

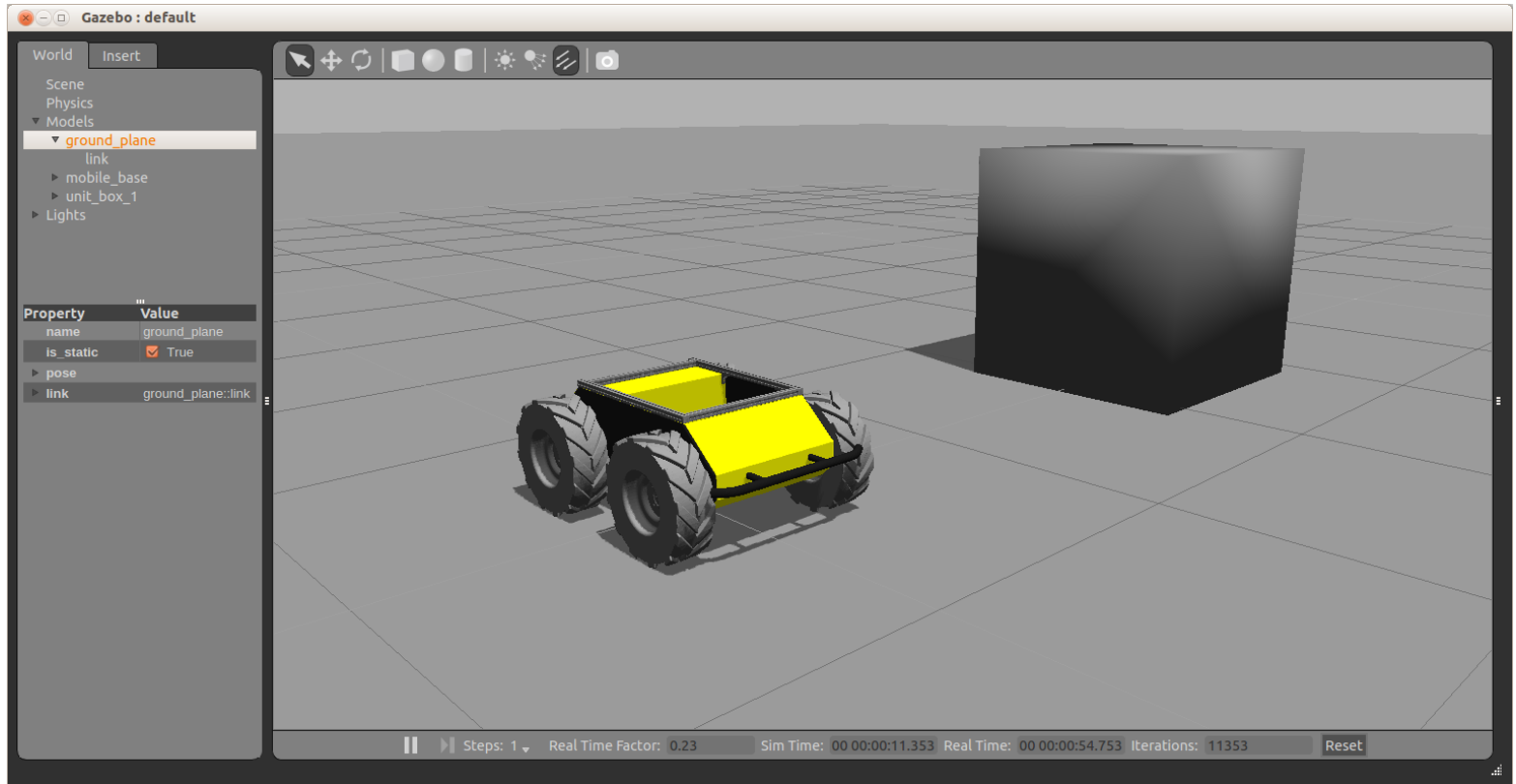
- The current range of drone simulators is relatively fragmented, with each autopilot project maintaining its own simulation backends. This is not only inefficient and a lot of parallel effort, but also undesirable for the open source ecosystem: it should be possible to combine a simulator with an autopilot stack freely, which will help to drive both software components individually to perfection.
- The PX4 project uses **Simulationkit**, an easy to use API to connect robotic simulators (not limited to drones) to software-in-the-loop (SITL) and hardware-in-the-loop (HITL) instances of the robot software. It currently supports MAVLink and ROS bindings.
- Simulationkit currently supports the jMAVSim and Gazebo simulators.



FLIGHT CONTROL & AUTOPILOT: SIMULATION



FLIGHT CONTROL & AUTOPILOT: SIMULATION



FLIGHT CONTROL & AUTOPILOT: GROUND CONTROL STATION

- QGroundControl provides full flight control and mission planning for any MAVLink enabled drone and configuration for ArduPilot or PX4 Pro powered vehicles. Its primary goal is ease of use for first time and professional users and is open source software.
- Support for managing multiple vehicles.
- Flight map display showing vehicle position, flight track, waypoints and vehicle instruments.
- Video streaming with instrument display overlays.
- Flight support for any vehicle which communicates using the MAVLink protocol.



Support for all vehicle types supported by PX4 Pro (multirotor, fixed-wing, VTOL, rover)

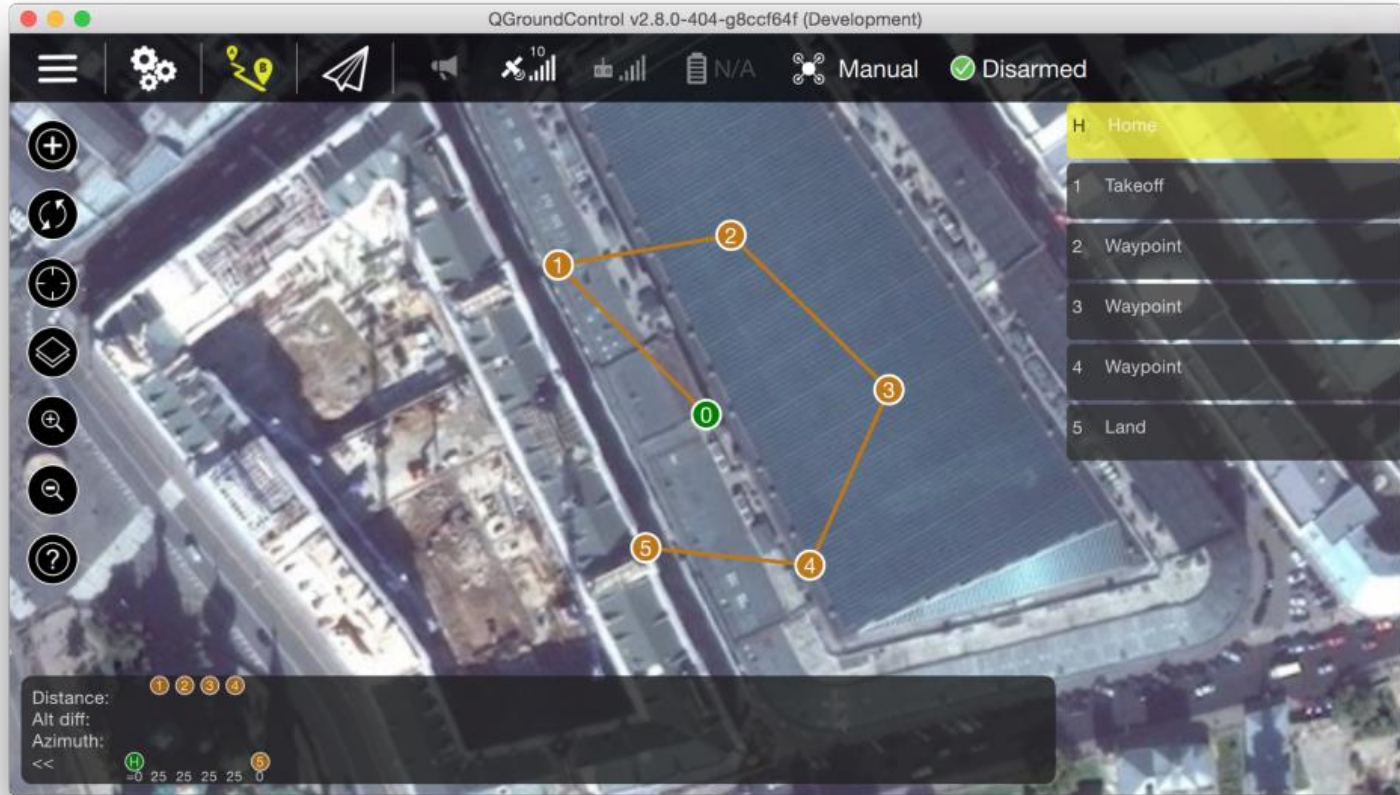


QGC runs on Windows, OS X, Linux, iOS and Android tablets

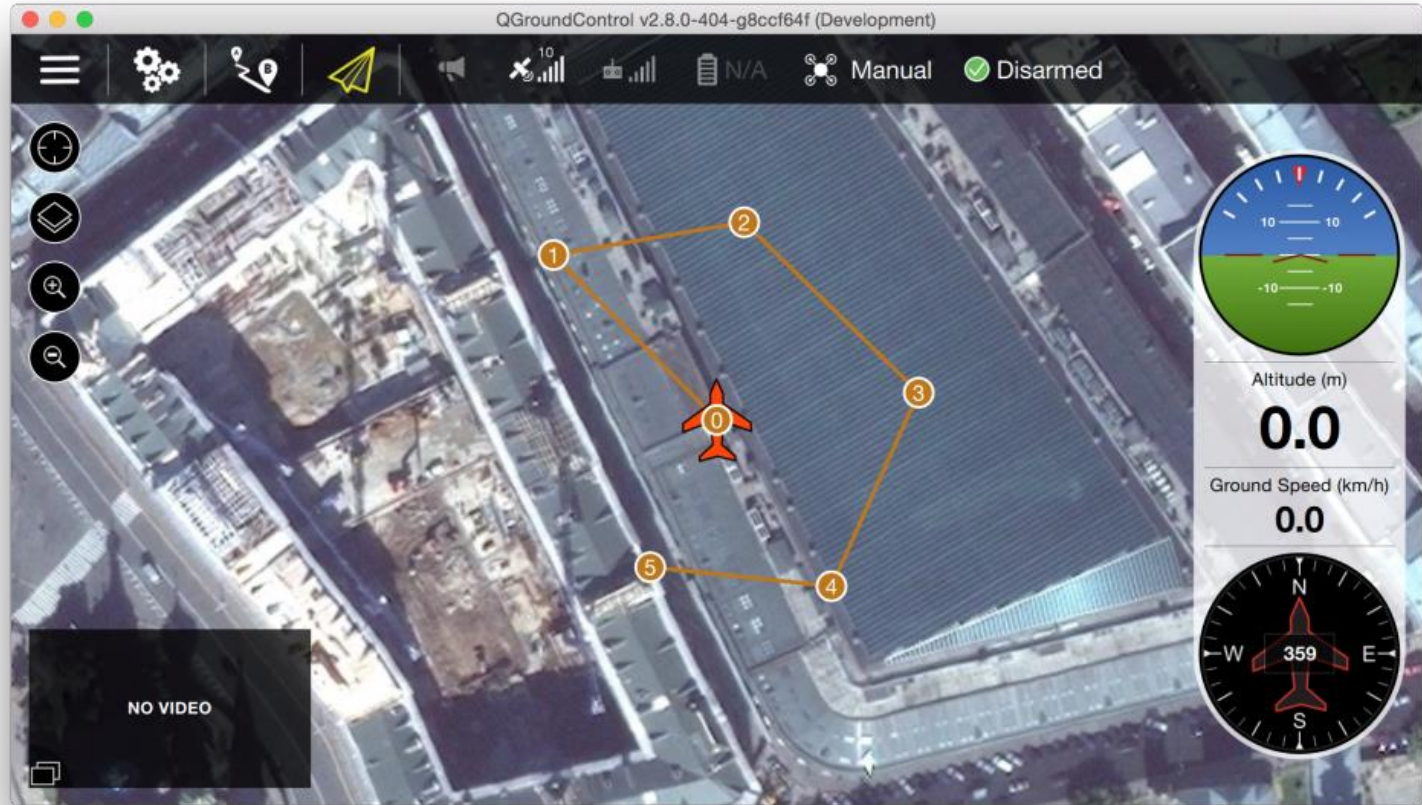


Supports multiple autopilots: PX4 Pro or any vehicle which communicates using the MAVLink protocol

FLIGHT CONTROL & AUTOPILOT: GROUND CONTROL STATION



FLIGHT CONTROL & AUTOPILOT: GROUND CONTROL STATION



A laptop is open on a wooden desk. The screen displays a code editor with a dark theme, showing a list of numbers and some text. To the right of the laptop, a tablet is visible, displaying a photo of a person. The background is slightly blurred, showing other desk items and cables. A large, semi-transparent yellow and black geometric shape is overlaid on the right side of the image.

USE CASE FEATURES

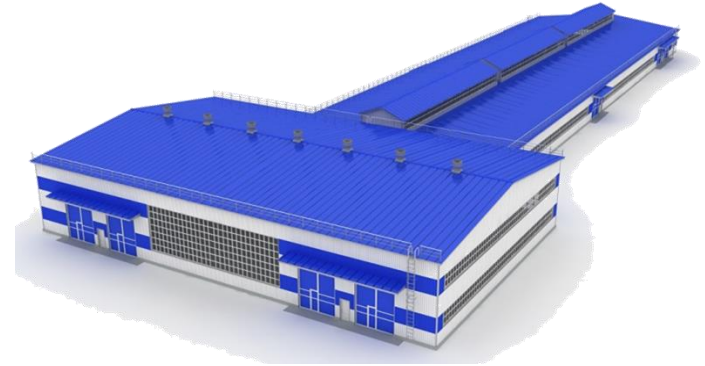
APPLICATION OVERVIEW

USE CASE: SYSTEM ARCHITECTURE



Operations Center

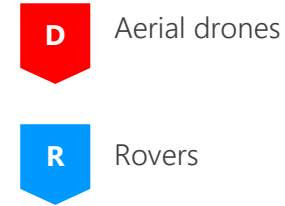
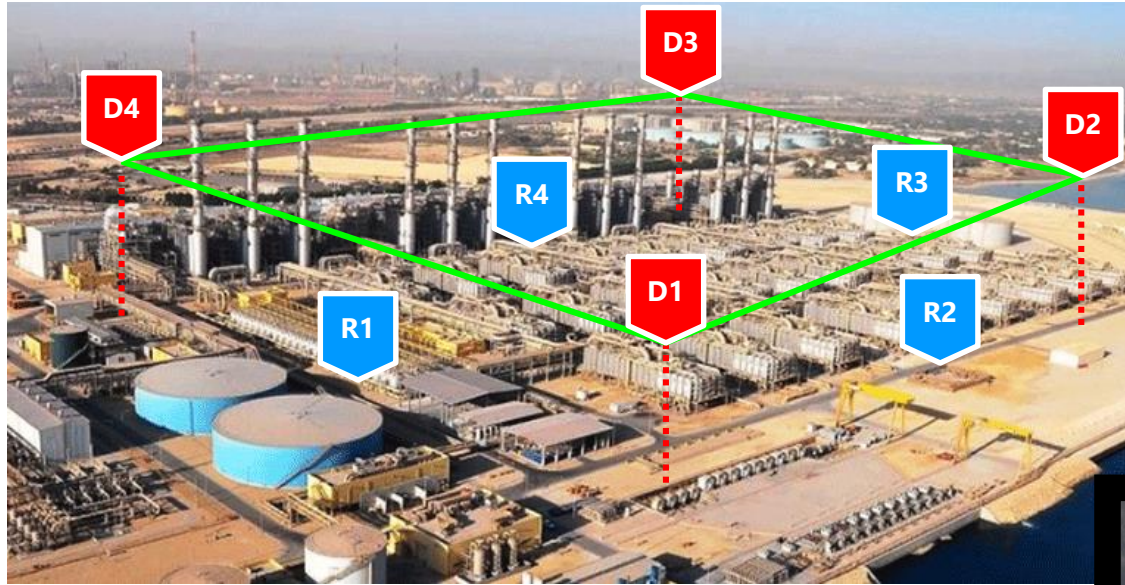
- Real-time video acquisition
- Log of all operations (GPS track, time and duration, images etc.)
- Real-time image processing (Computer Vision)
- Camera remote control from Operations Center (tilt and zoom)
- Remote control of dynamic waypoints



Patrol – Intervention

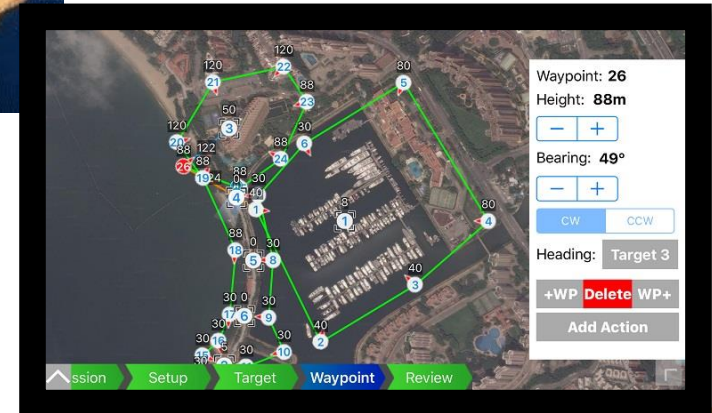
- Real-time video acquisition
- Autonomous flight
- Interaction with Operations Center by radio
- Wearable device for mission control (start, pause, stop mission, telemetry)

USE CASE: OPERATING SCENARIO



A single mission planner to define the area of intervention setting the limits, the path and the observation points.

Drones and rover are synchronized automatically, each device knows the position of others and communicates with the control center via 4G/LTE/WiFi



ADVANTAGES OF UAS SWARM

- The gathered information is used to help security personnel, first responders as well as rescue teams to conduct their mission efficiently.
- Both application scenarios share common requirements: sometimes a vast spatial area has to be inspected and information has to be provided to the stakeholders (security personnel, rescue teams, etc.) in real-time in case of an incident.
- Swarms can reduce the inspection/detection times compared to single UAV/rover applications. Especially in Search And Rescue (SAR) a single minute can decide between death and life.
- The inspection cycle time for surveillance can be reduced considerably enabling denser inspection.

UAS SWARM: FEATURES OF USE CASE

- At the core are multiple UAVs and multiple rovers that can act autonomously. They carry different sensors (VIS or IR cameras, microphones, gas sensors, etc.).
- Each vehicle either carries all sensor modalities or the sensor modalities are distributed among the vehicles.
- The vehicles carry intelligence and can communicate among each other (via WiFi, 4G or others).
- They operate as a self-organizing mixed team where particular tasks for each vehicle are not predefined at mission start but they negotiate the distribution of tasks among each other during mission execution.
- Such a swarm is highly adaptive to changes in the environment and can act dynamically. E.g. a ground robot may order a camera UAV to look for the best path, or a UAV finds something strange and orders UAVs with other sensors to check or asks if one of the rovers can move there to perform some action.
- Moreover, in contrast to full centralized control such a swarm can still operate even if the connectivity among vehicles or with a base station is sparse.

UAS SWARM: FEATURES OF USE CASE

- The swarm mission itself is defined in a central operation center in the beginning of the mission with a dedicated swarm definition tool (mission planner) that defines the goals and behavior of the swarm.
- The central station collects the sensor data and does sensor fusion and analysis in real time. The vehicles can also form a meshed ad-hoc communication network to improve communication performance.
- The central station carries according user interfaces to enable the operator to influence the swarm (e.g. tell that he wants to see a certain scene then the swarm automatically sends the closest vehicle to the scene, camera remote control) or documents the swarm mission including all GPS and sensor data.
- On top of this, members of intervention teams may access the swarm and its data directly via wearable devices and influence it.

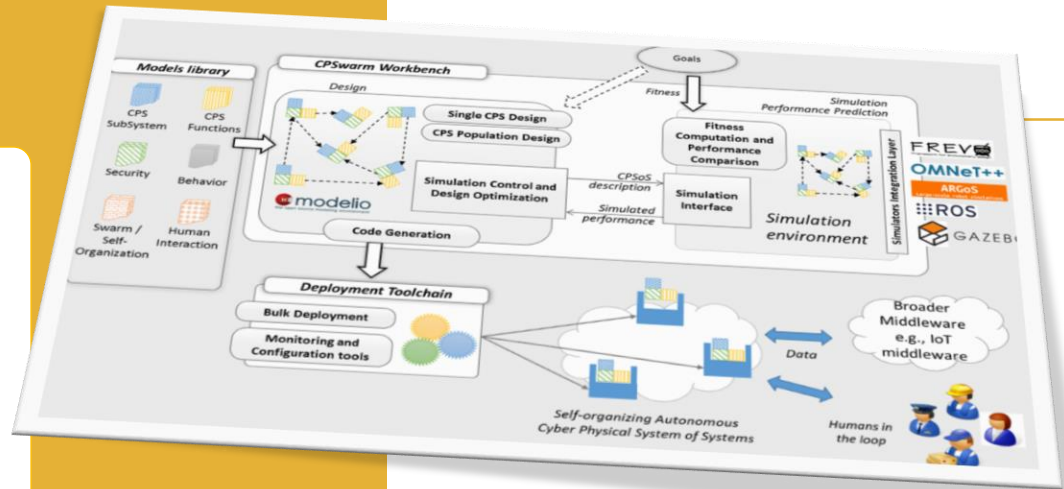


CPSwarm approach

Goal & concept

MAIN GOAL

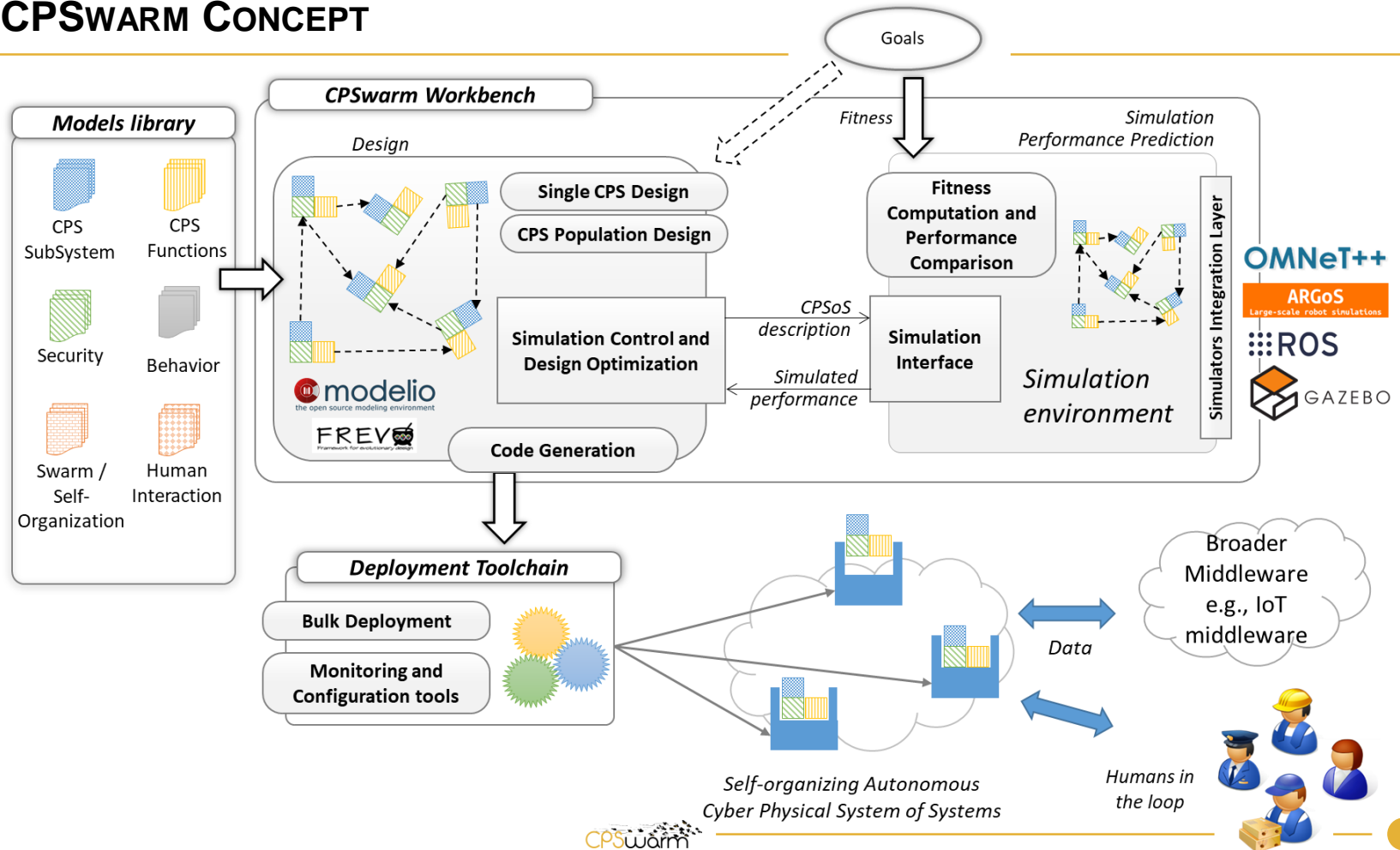
- The project aims at defining a **complete toolchain**, enabling the designer to:
 - Set-up **collaborative autonomous CPSs**;
 - Test the **swarm performance** with respect to the design goal
 - Massively **deploy solutions** of “**reconfigurable**” CPS devices and **CPSoS**.



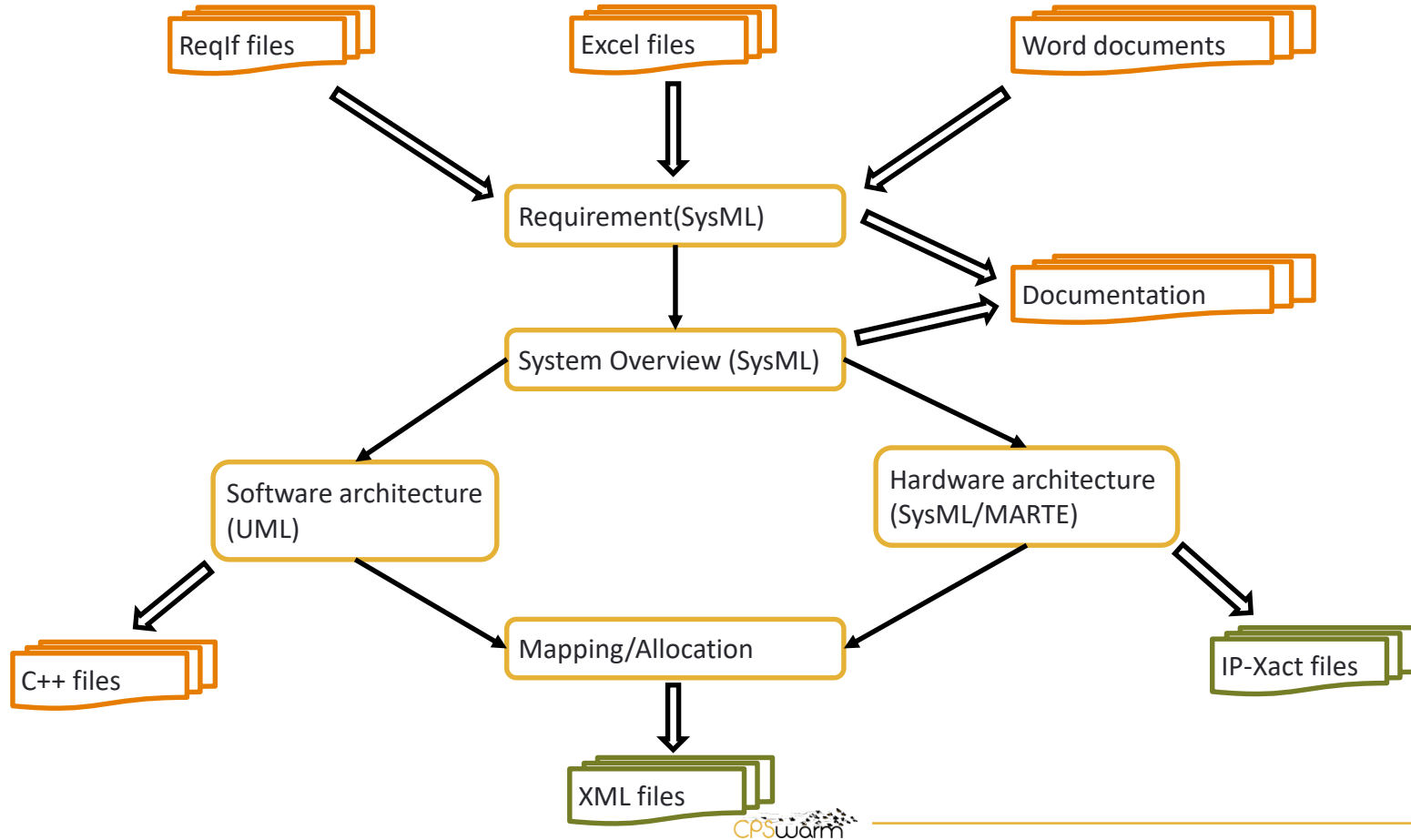
Design IDE and Workbench for CPS Swarms

CPSwarm offers a fully-fledged design and simulation environment, namely the **CPSwarm Workbench**, natively supporting iterative, **computer-aided model based design of CPSs**, with a particular focus on **swarms** of heterogeneous systems.

THE CPSWARM CONCEPT



MODELIO/CPS WORKFLOW



Objectives



O1: Drastically Improve support to **design** of **complex, autonomous CPS**



O4: Define a complete **library of swarm and evolutionary algorithms** for CPS design



O2: Provide a self-contained, yet extensible **library of re-usable models** for describing Cyber Physical Systems



O5: Establish **reference patterns and tools** for **integration** of CPS artefacts



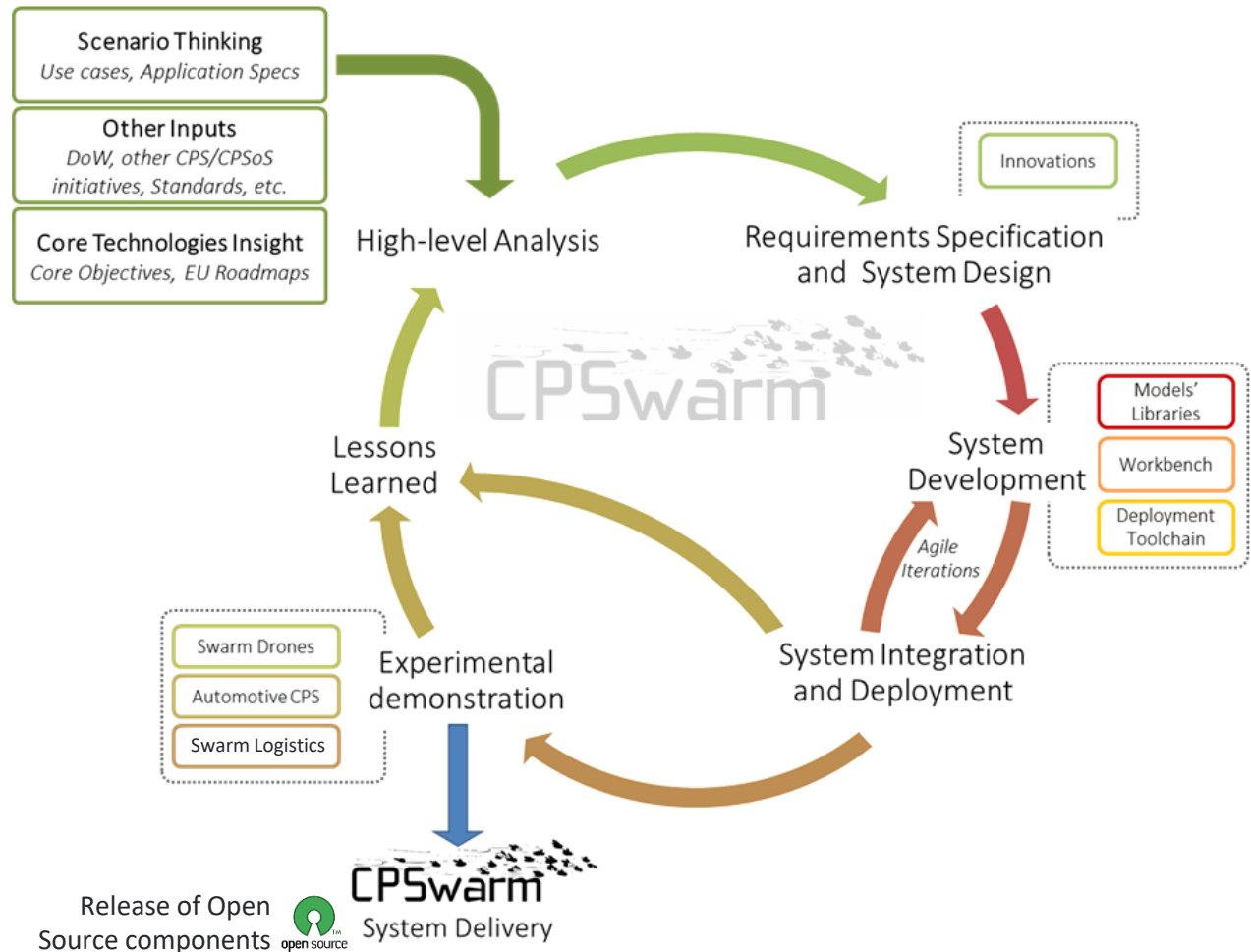
O3: Enabling a sensible **reduction in complexity and time of CPS development** workflow by automating deployment



O6: Address **real industrial needs** in CPS design, with a particular focus on the autonomous robotic vehicles, freight vehicles and smart logistics domain

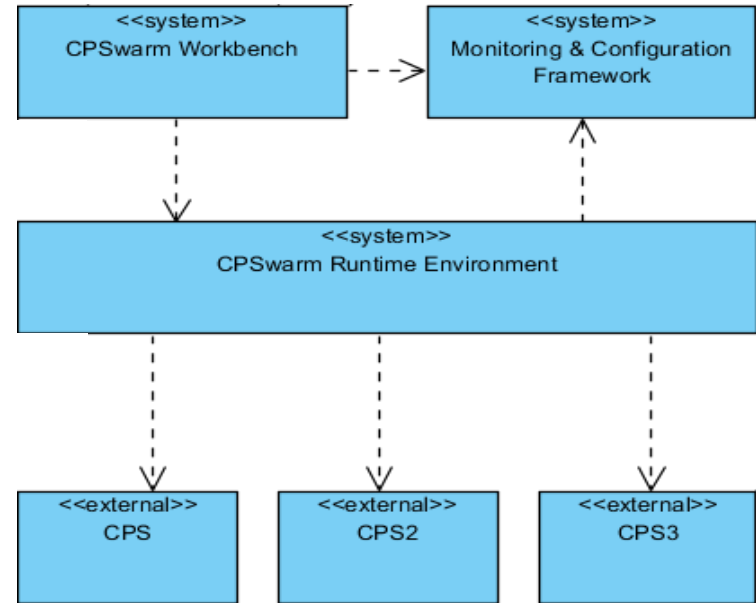
Methodology

- CPSwarm follows an **agile, iterative project methodology**, starting from the analysis of **three application scenarios** for swarms of CPSs.








CPSwarm Architecture – High-level Functional View

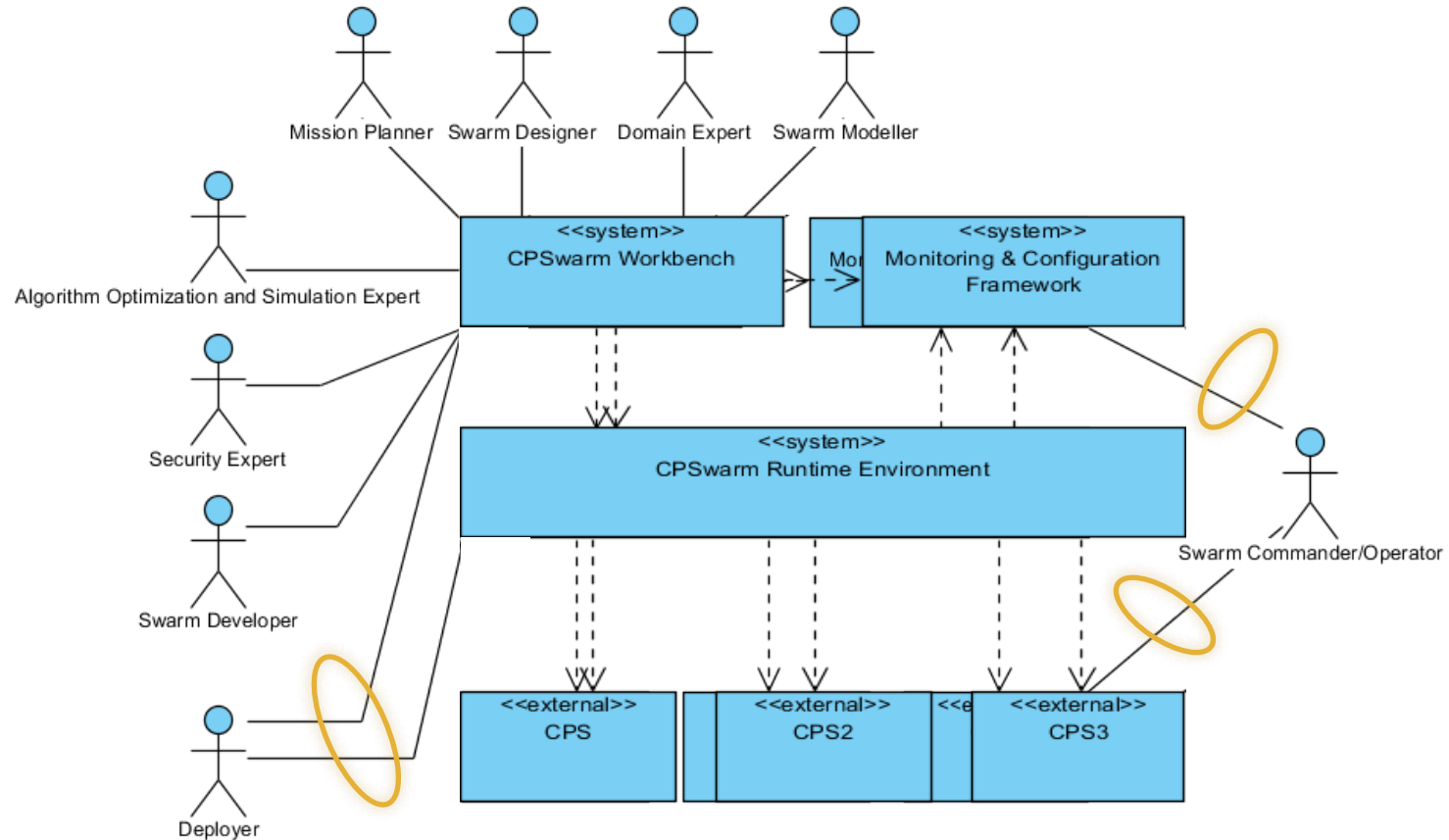
- The **CPSwarm Workbench** supports the core functions for **modelling, optimizing, simulating** and **deploying** swarms of CPS;
- The **CPSwarm Runtime environment** supports easier deployment on real-world platforms, providing functions and APIs designed to **decouple** the CPS **business logic** (the algorithms) from the **CPS infrastructure** (hardware and operating-system, for example), thus promoting **interoperability** and **reusability**;
- The **CPSwarm Monitoring & Configuration Framework** groups components located at the CPS side that enable real-time remote monitoring of the CPS operations and remote configuration of tunable parameters of designed swarm algorithms



CPSwarm Stakeholders

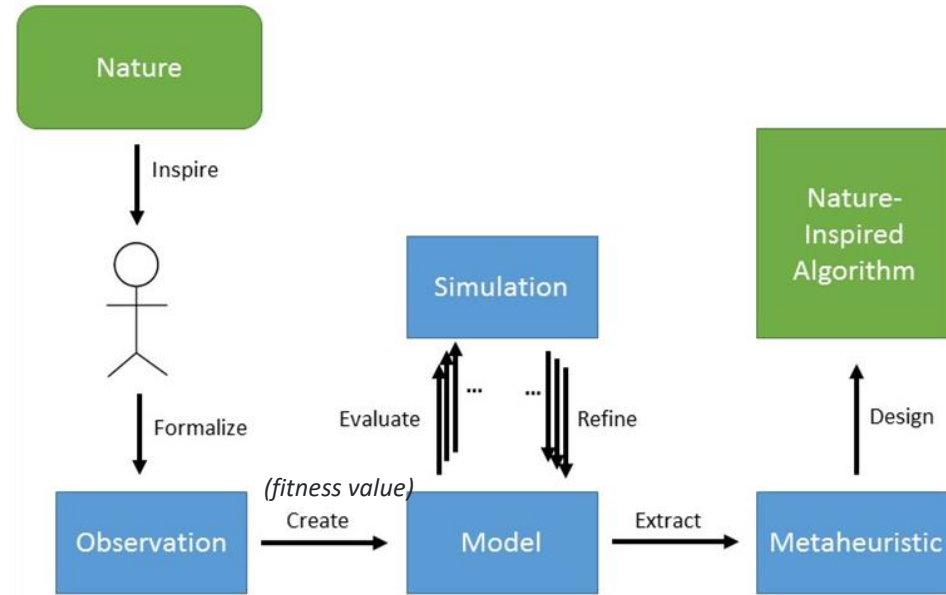
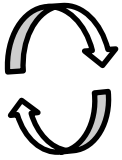
		Stakeholder	Description
Commercial Stakeholder 	{	Workbench Engineer	<i>A person, group or an organization responsible for the development and maintenance of the workbench</i>
		Mission Planner	<i>A person responsible for planning the mission. The mission includes problem definition, approach to solve the problem, environment description, mission parameters and mission success condition</i>
Modeller 	{	Swarm Designer	<i>A person responsible for designing the structure and behavior of the swarm based on the mission defined by the mission planner</i>
		Domain Expert	<i>A person, group or an organization who is an expert of the problem domain, also in terms of rules, regulations, limitations etc.</i>
		Security Expert	<i>A person, group or an organization responsible for providing expertise on safety and security of the swarm</i>
		Swarm Modeler	<i>A person who constructs the structure and behavioural model of the swarm</i>
Software Developer 	{	Algorithm Optimization and Simulation Expert	<i>A person or group who provides the expertise regarding the swarm algorithm. He decides the aptness of a certain algorithm given a specific swarm problem.</i>
		Swarm Developer	<i>A person or a group responsible for adding logic to the generated code. This code is later on deployed on each component of the swarm.</i>
Engineer 	{	Deployer	<i>A person or group responsible for deploying the code of the swarm.</i>
Operator 	{	Swarm Commander/Operator	<i>A person with the command control in his hand. He is responsible for directly manipulating the components of the swarm.</i>

CPSwarm Context view



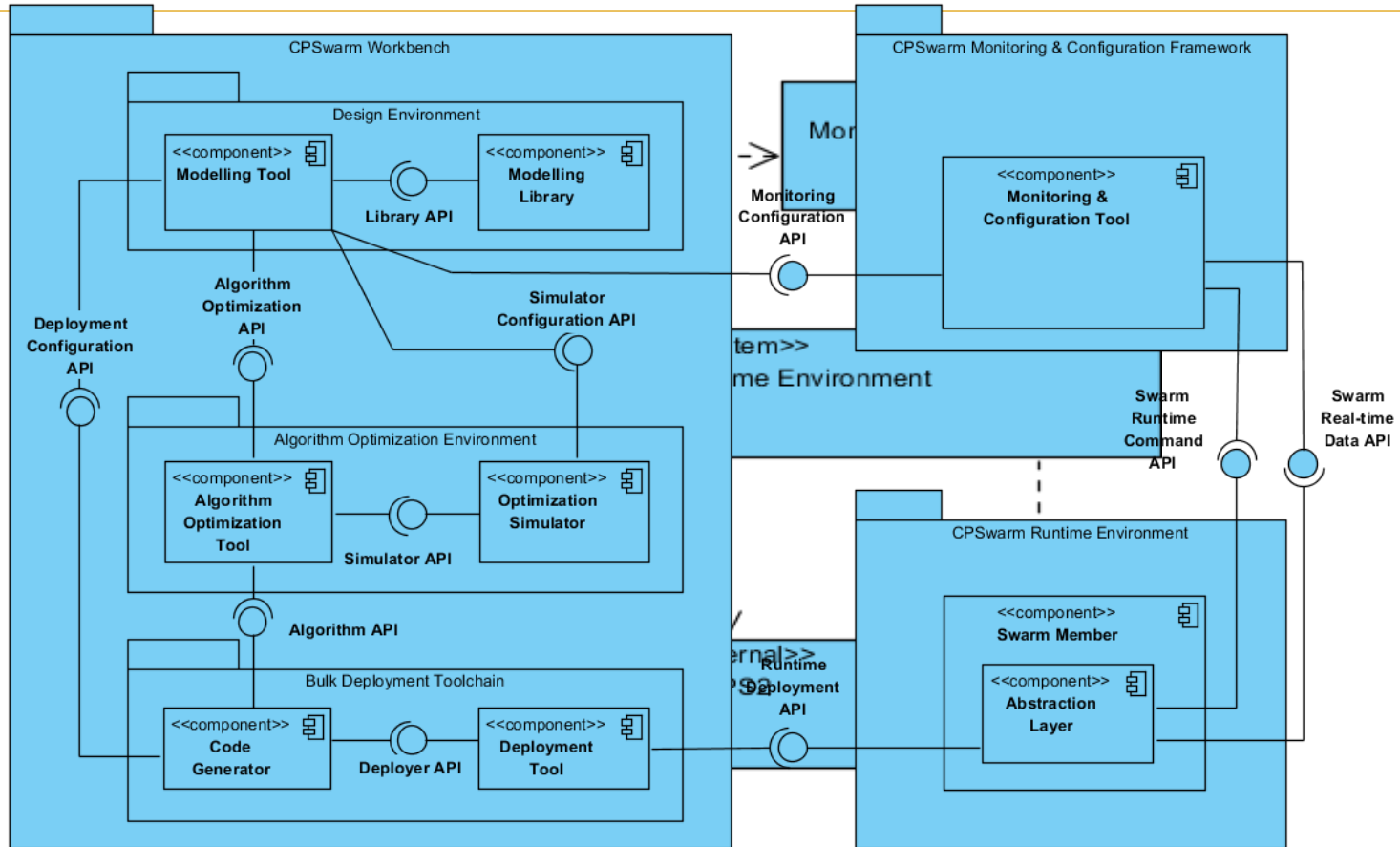
Swarm Intelligence Design

- In order to finally design a nature or bio-inspired **swarm intelligence algorithm**, a proper process can be followed
 - Analyze observations
 - Create a **swarm intelligence model** where the swarm is made of individual, simple **agents** communicating and cooperating without a central control
 - Perform **simulations** to assess the performance (i.e., solve complex tasks w.r.t. specific **fitness values**) that could be achieved with the **collective behaviour** emerging from the interaction among the different agents
 - Extract the researched algorithm



Adapted from H. R. Ahmed and J. I. Glasgow, "Swarm Intelligence: Concepts, Models and Applications," Kinston, Canada, Queen's University, School of Computing Technical Reports, 2012.

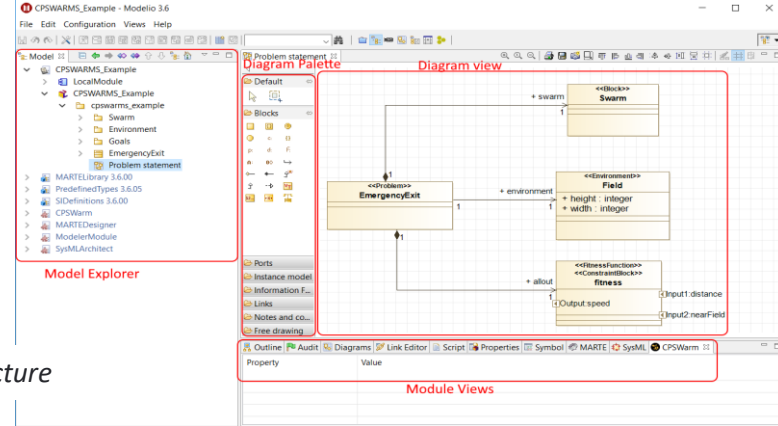
CPSwarm Architecture – high level functional view



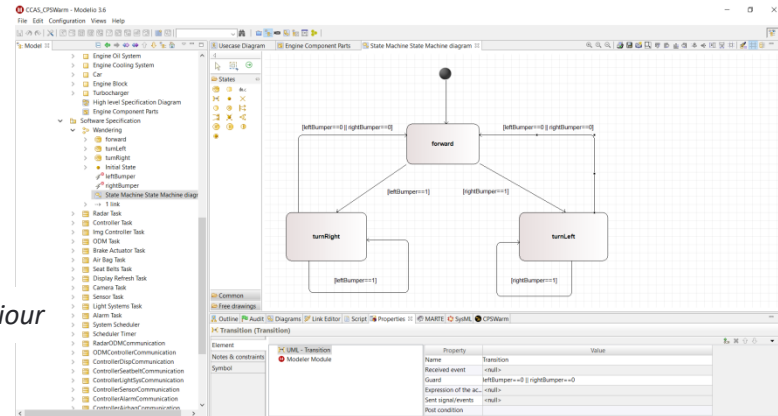
Design Environment – Modeling Tool

It integrates a **GUI** offering functions to model the swarm structure, behavior, environment and other necessary parameters

- It provides an easy way for **Swarm Designers** to design a swarm without having profound expertise in programming and/or hardware specific knowledge
- Block-based design UIs and tools for identifying and **composing single CPS systems**
- Tools to **compose populations of (heterogeneous) CPSs**
- It exploits **modelling languages** among the available standards including SysML and MARTE



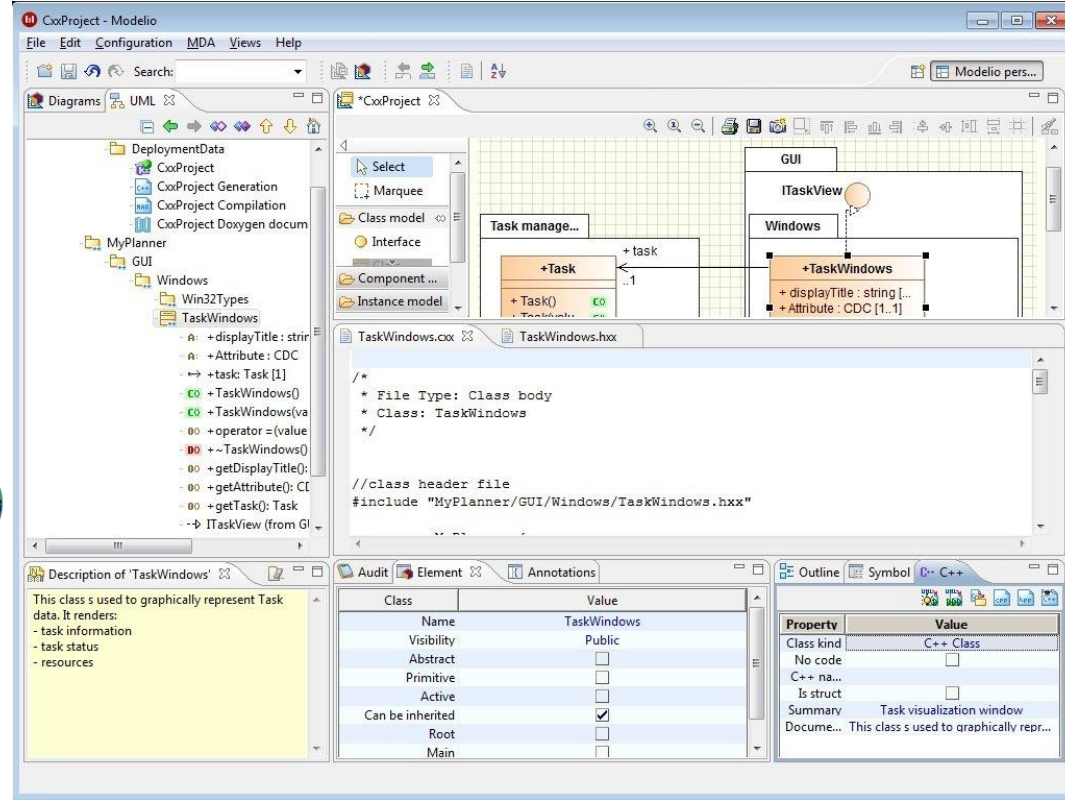
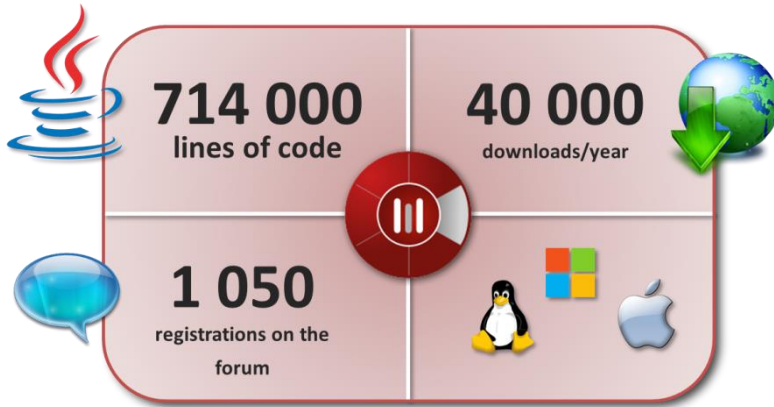
Structure



Behaviour

Modelio for System Engineering

- UML editor with 20 years' history
 - SysML, MARTE, BPMN
 - Code generation
 - Documentation
 - Available under open source at Modelio.org

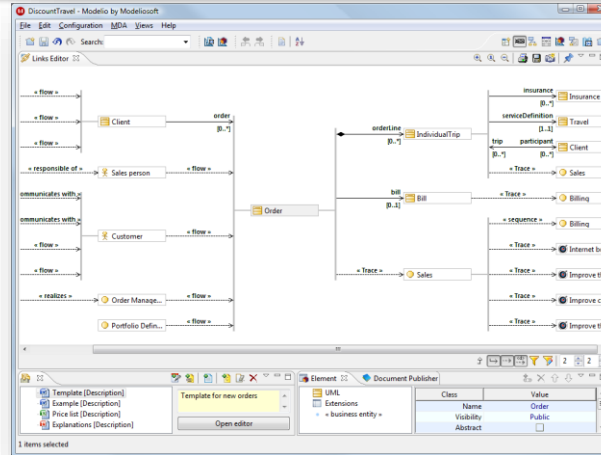


Modelio 3

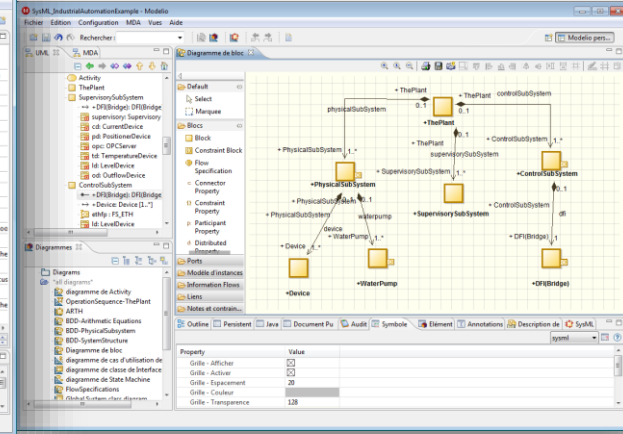
Modelio System Architect Solution

Dedicated to System architects modelling with SysML, UML or BPMN and carrying out Requirements based analysis

- Modelling with UML, SysML and BPMN
- Requirements Modeling
- Tabular editors
- Import/export MS Excel & Word
- Embedded Systems modelling via MARTE
- Traceability Editor
- Impact Analysis
- Document Generation
- Support for Collaborative activities (Constellation, SVN)
- Automatic diagrams creation
- Customisable, interfaces to external tools



**Traceability
Editor**



**Dedicated
SysML editor**

Design Environment – Modelling Library

*A library collecting reusable CPS descriptions, swarm behavior algorithms, security guidelines etc. that can be properly adjusted, modified or extended
It enables high **reusability** and **interoperability** of core functions adopted in swarm development*



Categories of modelling libraries

Agent Library



- **Local status** (status of the agent including e.g., available resources but also its position)
- **Behaviour** (application logic e.g., collect sensors measurements and send data)
- **Physical aspects** (hardware characteristics, sensors, actuators)
- **Security** (models for threat analysis and main countermeasures)
- **Human interaction** (direct or mediated)

Environment Library



- **2D/3D map** of the environment (occupancy grid map, i.e. free space and obstacles expressed as a bitmap file)
- **Size** of the environment (width and height expressed in number of grid cells)
- **Resolution** (expressed in number of grid cells per meter)

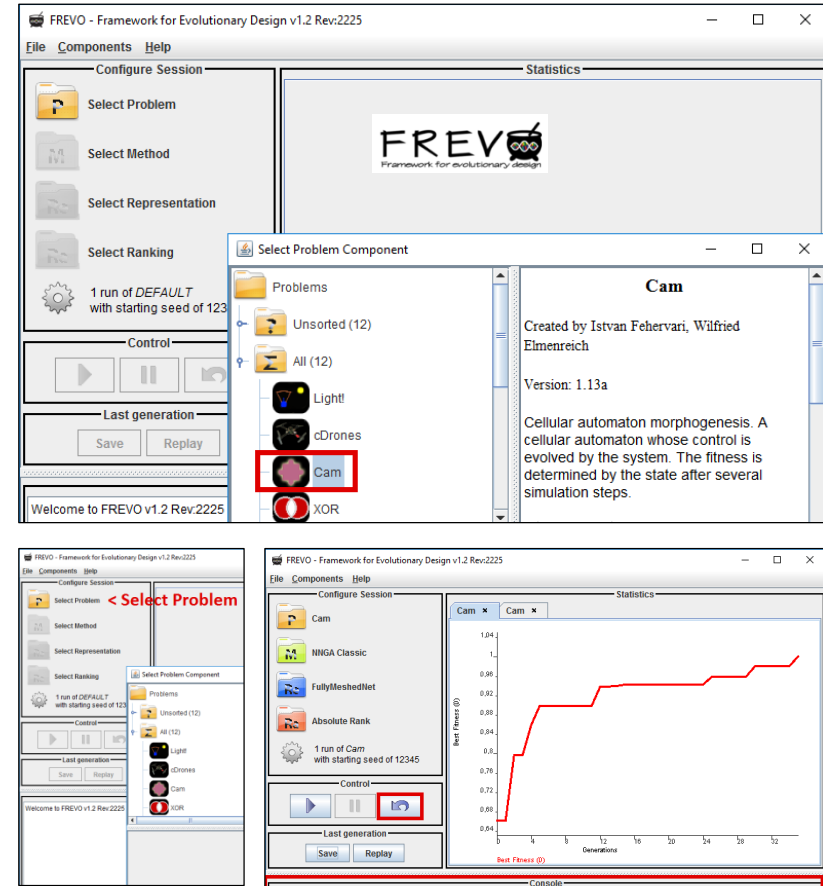
Goal Library



- One or multiple **fitness values**
- **Calculation specification**, actually incorporating parameters from different models

Algorithm Optimization Environment – Optimization Tool

- It adopts **evolutionary methods** to automatically optimize the algorithm of individual swarm members that collectively contribute to a target swarm emergent behaviour.
- It supports agent modelling and **evolvable representations** (e.g., Artificial Neural Network) of the agent controller.
- An **iterative heuristic search** is applied to find an optimized configuration of the controller for a given CPS w.r.t. a system level optimization measure, called **fitness value**.
- The controller is evaluated by the Optimization Simulator by performing a statistically significant number of **simulations**.



Algorithm Optimization Environment - Optimization Simulator

- It is used to *evaluate the performance* of a generated controller algorithm/module.
- At each generation of the evolutionary optimization, it executes the current controller in a predefined environment.
- Depending on the problem to be solved, different simulators can be used. Relevant requirements have been identified
- easy of use, flexibility, extensibility, **scalability**, tunable granularity



Simulation results are exploited to compute a *fitness score*, allowing the Algorithm Optimization Tool to further refine the controller.

Simulation Tools under evaluation

2D

Simulation Engine	License	Language formats /	OS
Stage	GPL v2.0	C++, Configurations in plain text	Linux, Windows
TeamBots	Free for education and research	Java, configuration in source code or plain text files	Linux, Windows, MacOS
Swarm	GPL v2.0	Java – Objective-C	Linux, Windows, MacOS, Solaris
MRSim	All rights reserved	Matlab	
STDR	GPL v3.0	C++, configuration in XML and YAML	Linux
Rossum Playhouse	GPL v2.0 / MIT	Java	
MobotSim	All rights reserved	Visual Basic	Windows

3D

Simulation Engine
Gazebo
ARGoS
Webots
Swarmbot3D
MuRoSimF
DPRSim
Mission Lab
MORSE
SimSpark
V-REP
Breve
Simbad
Marilou
jMAVSim
peekabot

Enabled Research Activities



MODELING

Research work and development of functional, local behavior, global behavior and human interaction models



SWARM ALGORITHMS

Definition and implementation of swarm algorithms and models for tools to design swarm algorithms



SECURITY THREATS AND ATTACKS

Describing, defining and developing models to evaluate the security level of the overall CPS



Bulk Deployment Toolchain

*The Bulk Deployment Toolchain is responsible for the **generation** of platform-specific source code and the **deployment of code** into designated runtime environments. Its extensibility characteristics enable easier integration of diverse CPS platforms and support extensions by third party developers*



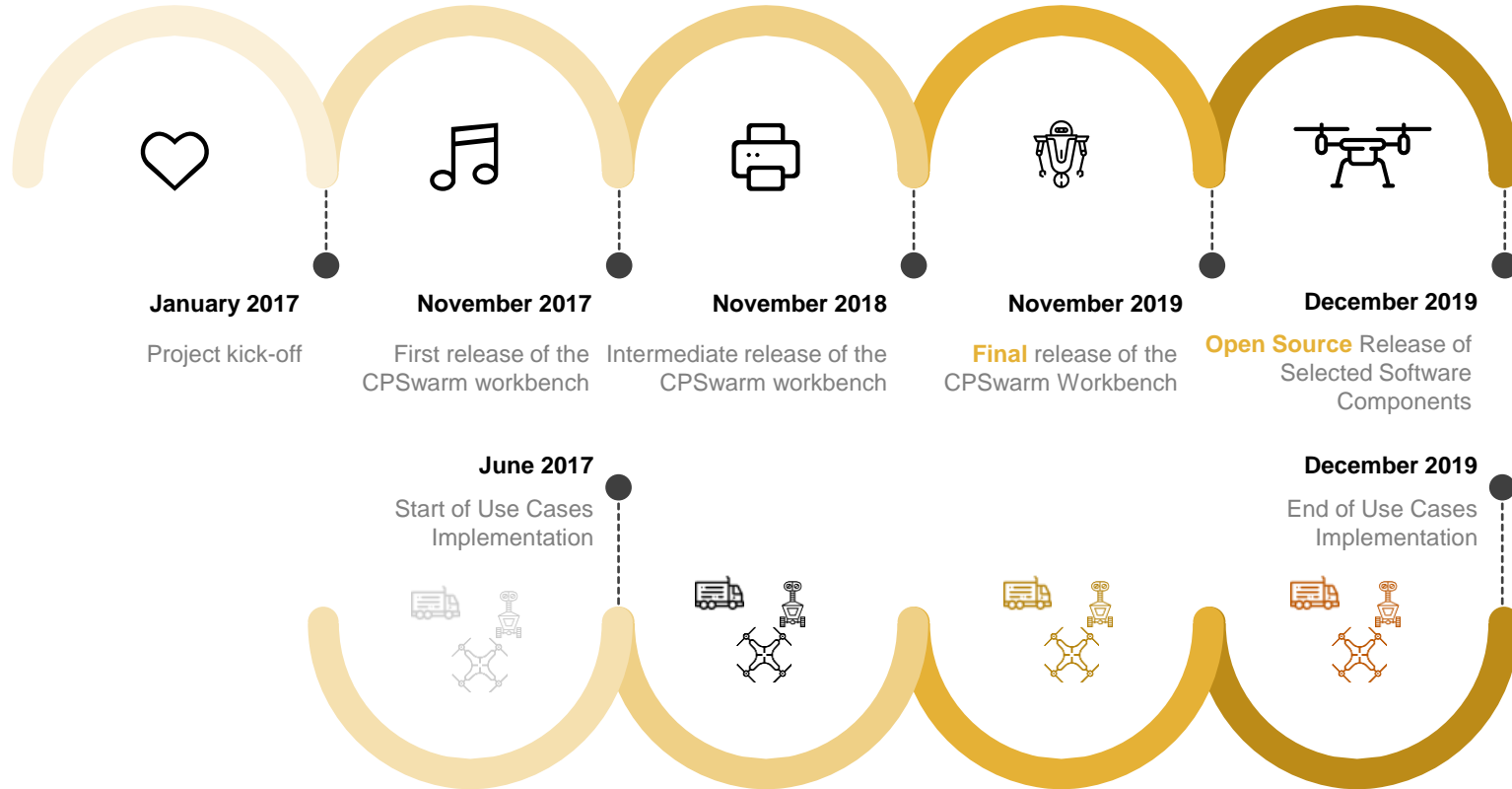
Bulk Deployment Toolchain

- **Code Generator** – two generation patterns considered
 - **Template based**
 - Lower complexity (spread between the template and the generator, using a XSLT like approach)
 - Can be cascaded (generated code can be used as input for other templates)
 - **Programmatic generation**
 - Higher complexity (mainly associated to the generator)
 - Cannot be cascaded but it is more flexible (no need for “pre-compiled” pieces of code)
- **Deployment Tool** – the generated code can be deployed on different platforms, following two main strategies
 - **Direct update**
 - **Over-the-air (OTA) update**

```
<scxml xmlns="http://www.w3.org/2005/07/scxml" version="1.0" initial="forward" name="Wandering">
  <state id="forward">
    <transition event="leftBumper" cond="leftBumper==1" target="turnRight"/>
    <transition event="rightBumper" cond="rightBumper==1" target="turnLeft"/>
  </state>
  <state id="turnRight">
    <transition cond="leftBumper==1" target="turnRight"/>
    <transition cond="leftBumper==0 || rightBumper==0" target="forward"/>
  </state>
  <state id="turnLeft">
    <transition cond="rightBumper==1" target="turnLeft"/>
    <transition cond="leftBumper==0 || rightBumper==0" target="forward"/>
  </state>
</scxml>
```

```
:SM($scxml.name)
{
  enum STATES
  {
    #foreach($target in $scxml.targets.keySet())
      $target#If( $foreach.hasNext ),#end
    #end
  }
  FSM_START($scxml.initial);
  FSM_BGN
  {
    #foreach($target in $scxml.targets.keySet())
      FSM_STATE($target)
      {
        FSM_CALL_TASK($target)
        FSM_TRANSITIONS
        {
          #foreach($transition in $scxml.targets.get($target).transitionsList)
            FSM_ON_CONDITION($transition.cond, FSM_NEXT($transition.next));
          #end
        }
      }
    #end
  }
}
```

Main Milestones





**ASK MORE
QUESTIONS**

THANKS!
ANY QUESTIONS?

CONTACTS



Alessandra Bagnato
Softeam R&D



Alessandra.bagnato@softeam.fr



@alebagnato



www.softeam.fr



<http://www.cpswarm.eu>



Follow @CPSwarm_EU



Coordinator



Istituto Superiore Mario Boella

Partners



Robotnik

SOFTEAM Cadextan

Lakeside Labs

TTTech
Ensuring Reliable Networks

ALPEN-ADRIA
UNIVERSITÄT
Klagenfurt



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731946.

