ComputationWorld2018

CPSwarm

Tutorial

Melanie Schranz (Lakeside Labs)
schranz@lakeside-labs.com

Lakeside Labs and the University of Klagenfurt

AGENDA

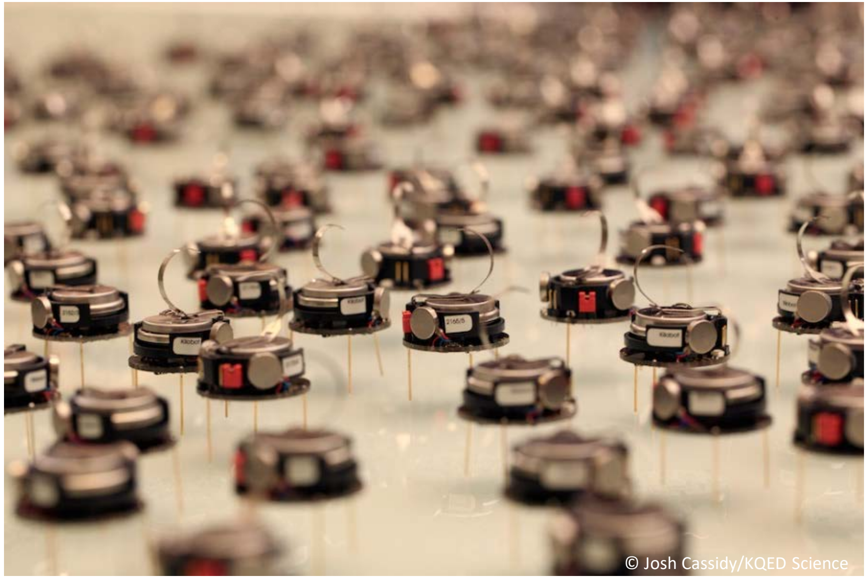
CPSWARM PROJECT

MODELING CPSS AND SWARMS OF CPSS

LIVE TUTORIAL/DEMO OF FREVO

Vision

Interactions amongst CPS might lead to new behaviors and emerging properties, often with unpredictable results. Rather than being an unwanted byproduct, ***these interactions can become an advantage if explicitly managed*** *since early design stages.*



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 R**esearch and **I**nnovation **A**ction (**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 ≈ 16 FTE)

The CPSwarm Consortium

Coordinator

IT — ISMB, Istituto Superiore Mario Boella

DE — Fraunhofer FIT

ES — Robotnik

AT — Lakeside Labs, SELF-ORGANIZING NETWORKED SYSTEMS

AT — TTTech, Ensuring Reliable Networks

AT — ALPEN-ADRIA UNIVERSITÄT KLAGENFURT I WIEN GRAZ

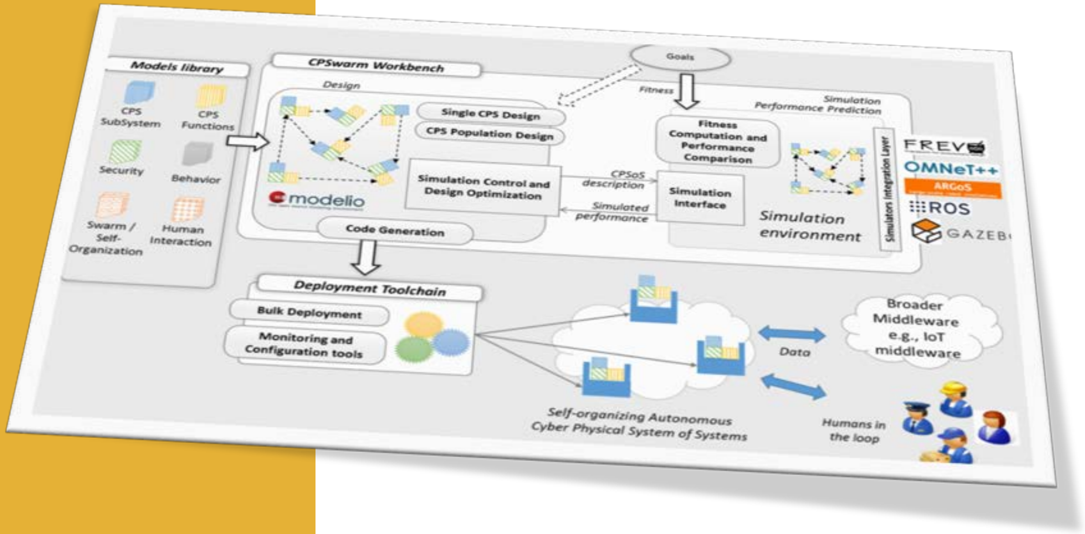
HU — SEARCH-LAB, SECURITY EVALUATION ANALYSIS AND RESEARCH LABORATORY

IT — DigiSky

FR — SOFTEAM Cadextan

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.

Objectives

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

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

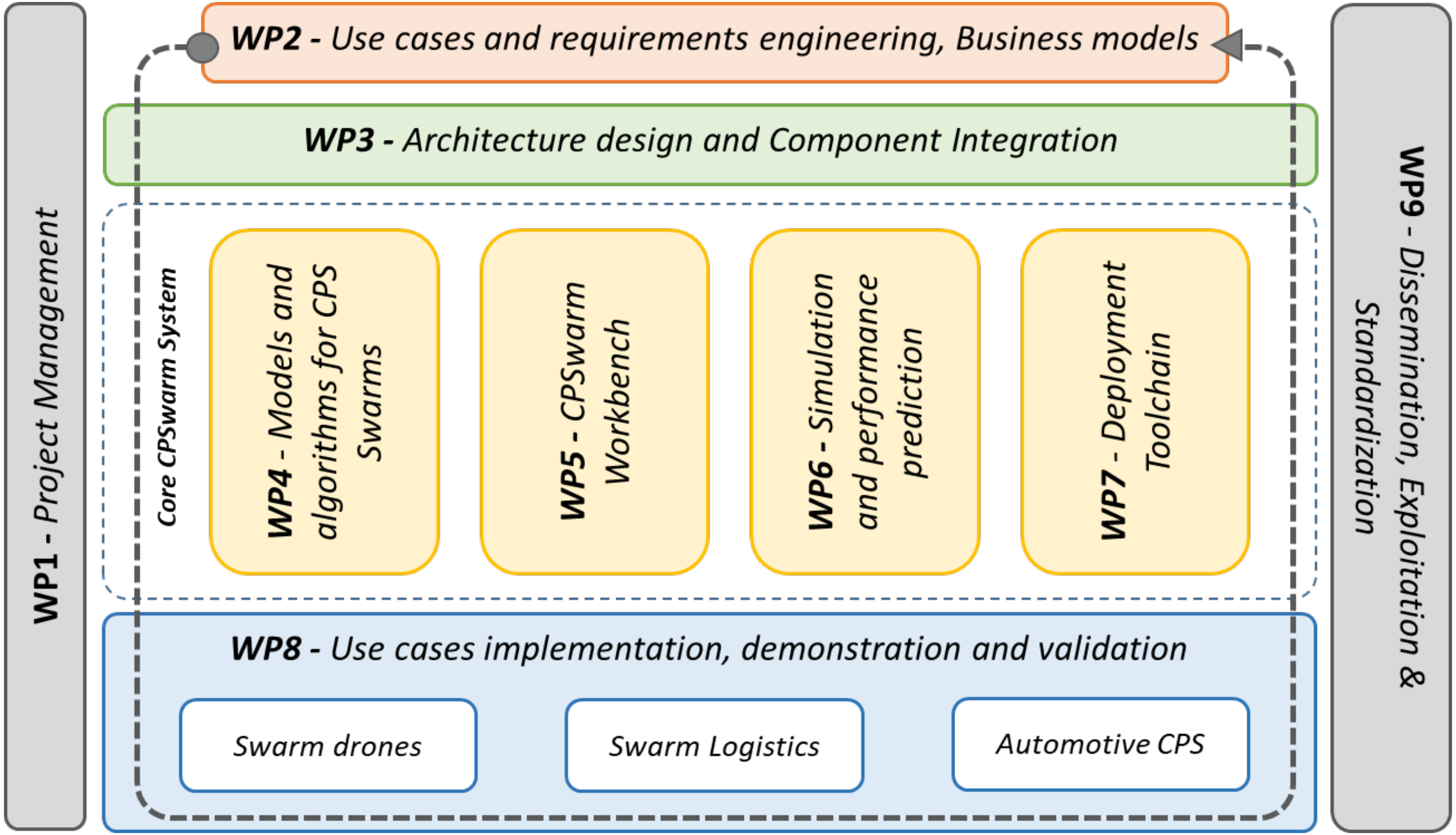
O3: Enabling a sensible reduction in complexity and time of CPS development workflow by automating deployment

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

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

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

CPSwarm Work Packages



Application Scenarios

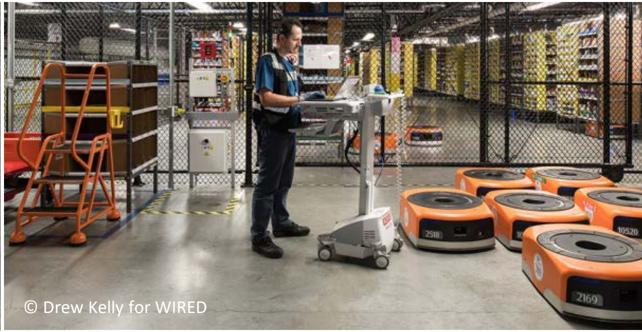
Three reference Application Scenarios drive the collection of requirements for the development of the **complete CPSwarm toolchain** *supporting the engineering and deployment of CPS swarms*


Phantom_Glacier: Image courtesy DJI

Swarm Drones

© dailymail.co.uk

Automotive CPS


© Drew Kelly for WIRED

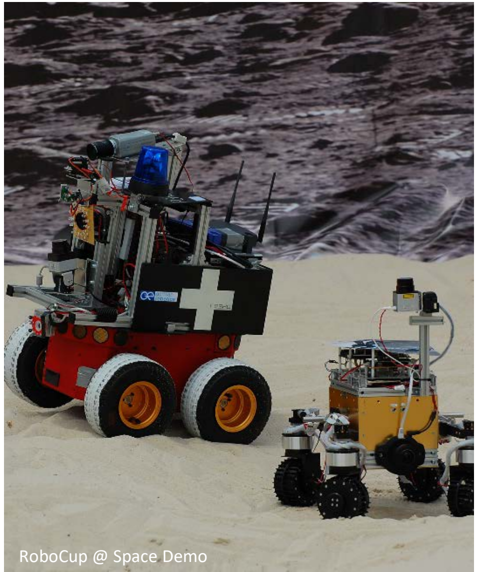
Swarm Logistics Assistant

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.

Phantom_Glacier: Image courtesy DJI

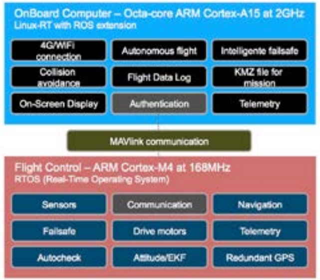


RoboCup @ Space Demo

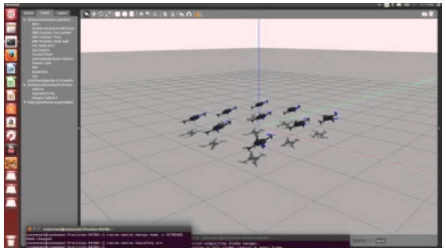
© MAXSUR

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).



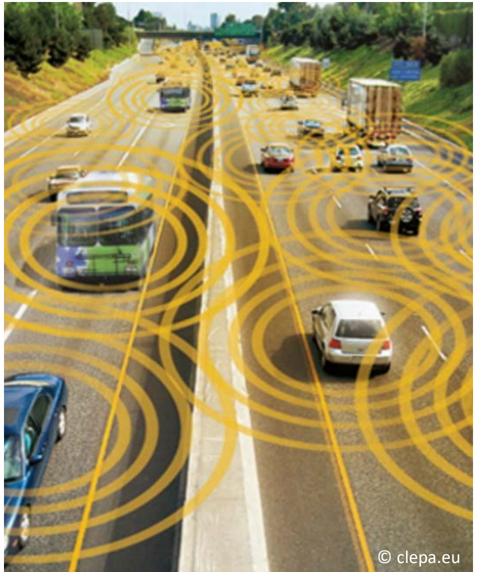
- The **model** of a drone, including HW characteristics, physical aspect and behaviour, can be created using **SDF** i.e., an XML format that describes objects and environments for robot simulation, visualization, and control.

Automotive CPS

- 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.

© dailymail.co.uk

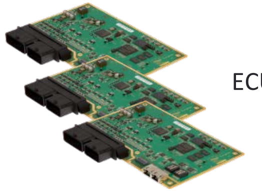

© clepa.eu


© Daimler

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.

Automotive Fog node



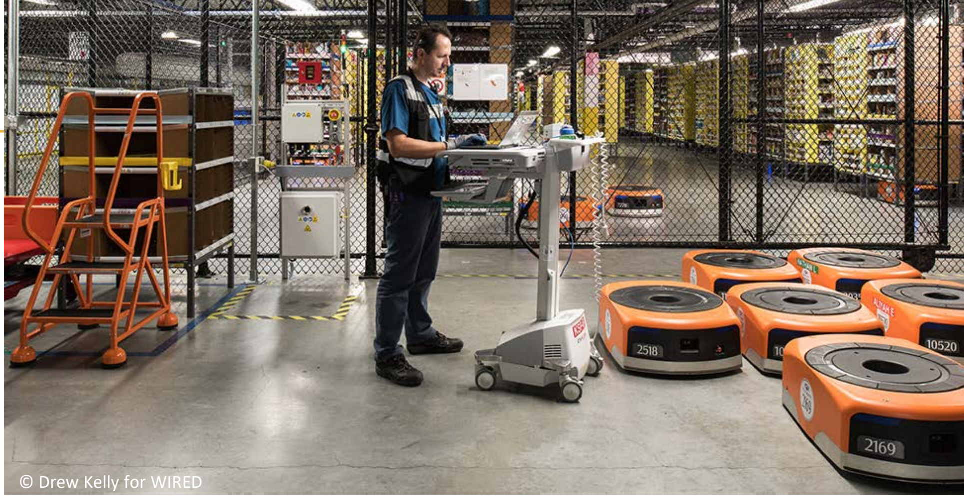
ECUs

Deterministic Ethernet Switch

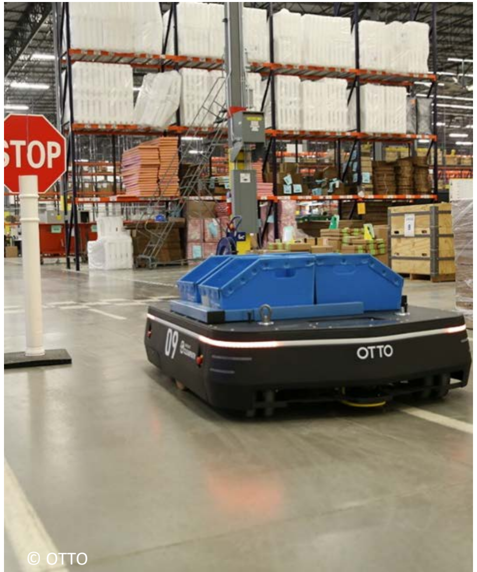
Swarm Logistics Assistant

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



© Drew Kelly for WIRED



© OTTO

© Fraunhofer IML

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 connected

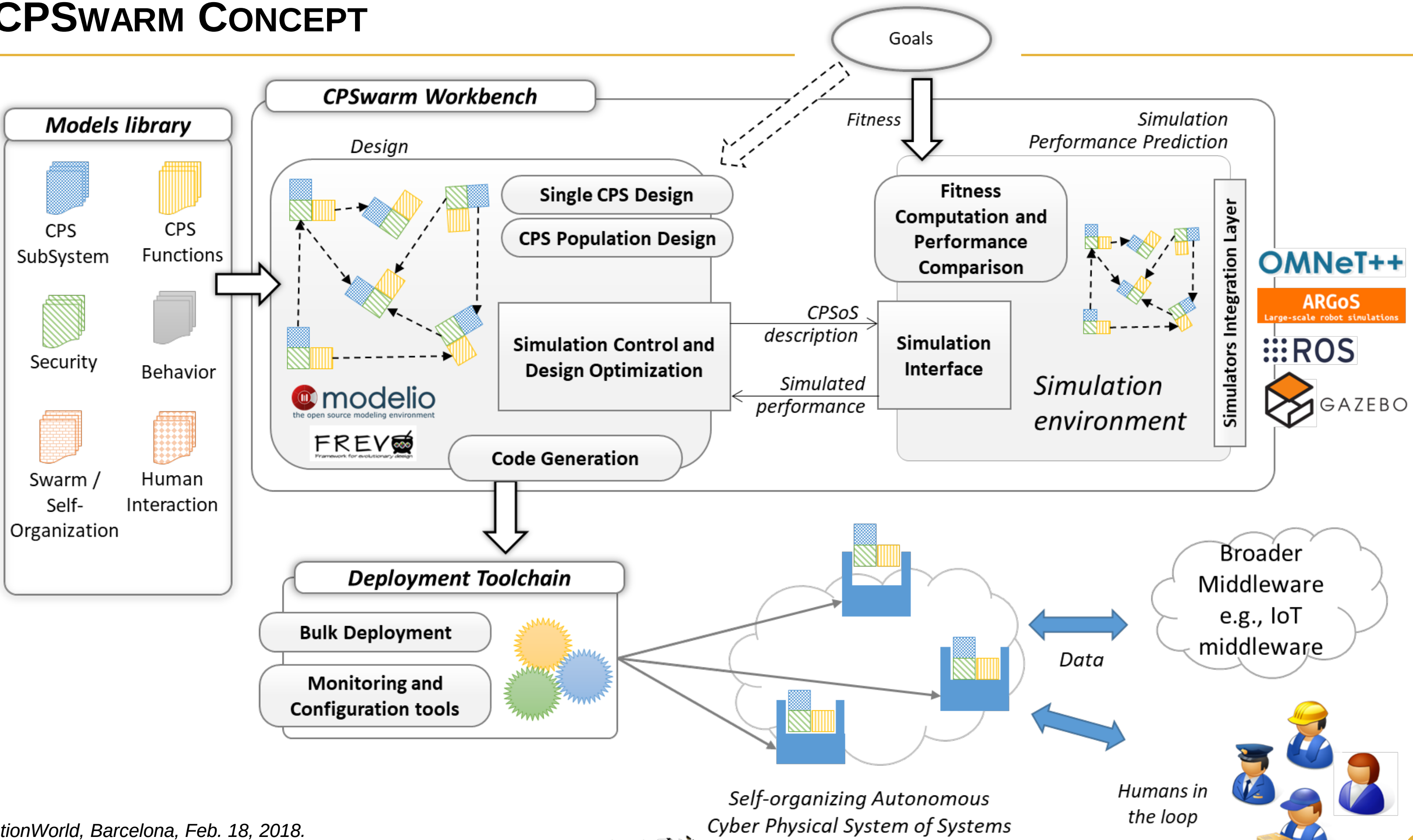
  and a hierarchy-based modelling to describe any kind of robot
- **Robots simulation** is enabled by ROS and Gazebo

Turtlebot 2

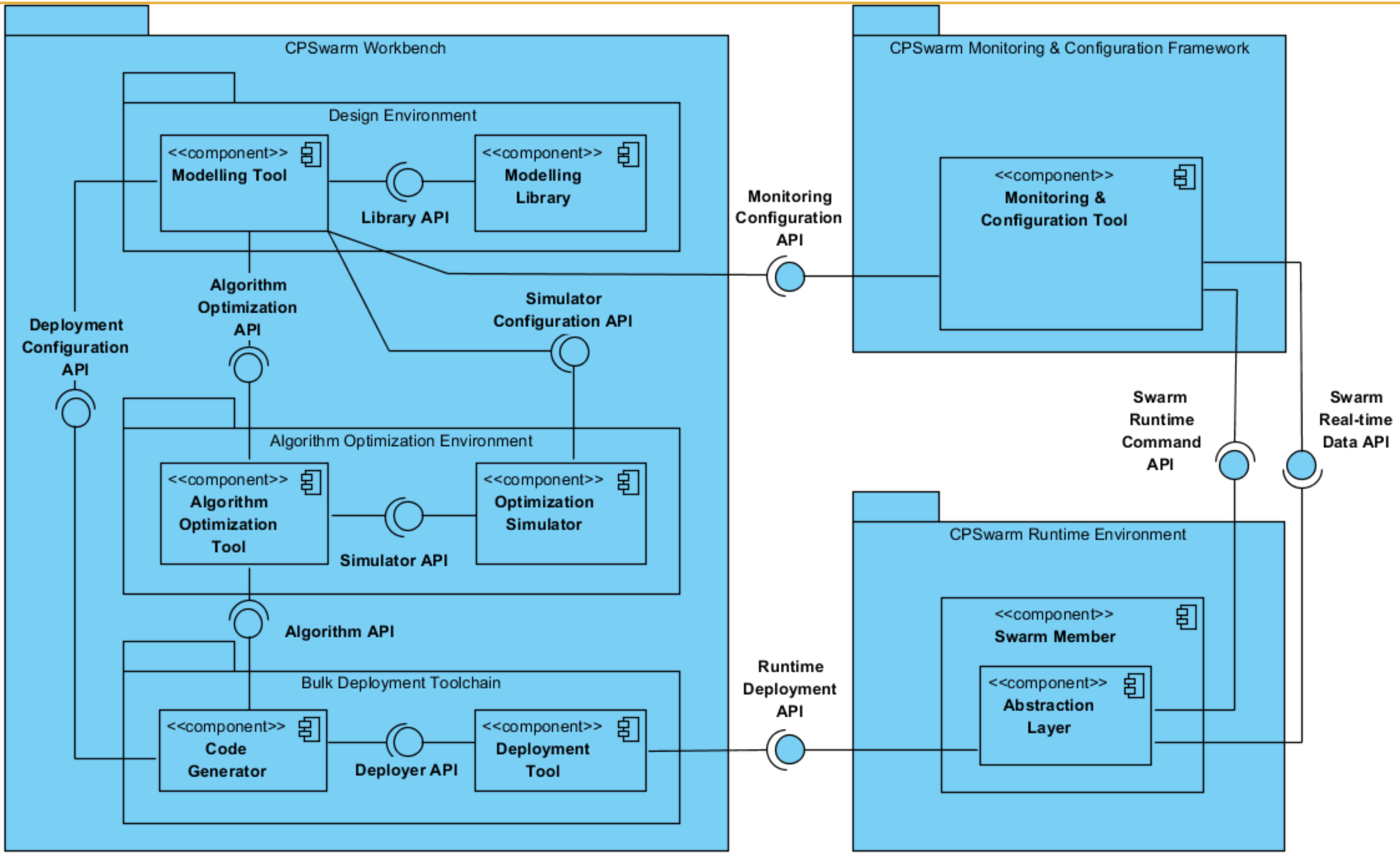
AGVS

The CPSwarm Workbench

THE CPSWARM CONCEPT

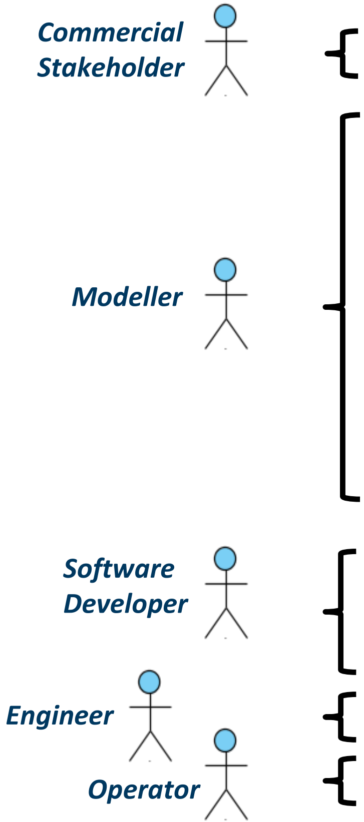


CPSwarm Architecture



CPSwarm Launcher

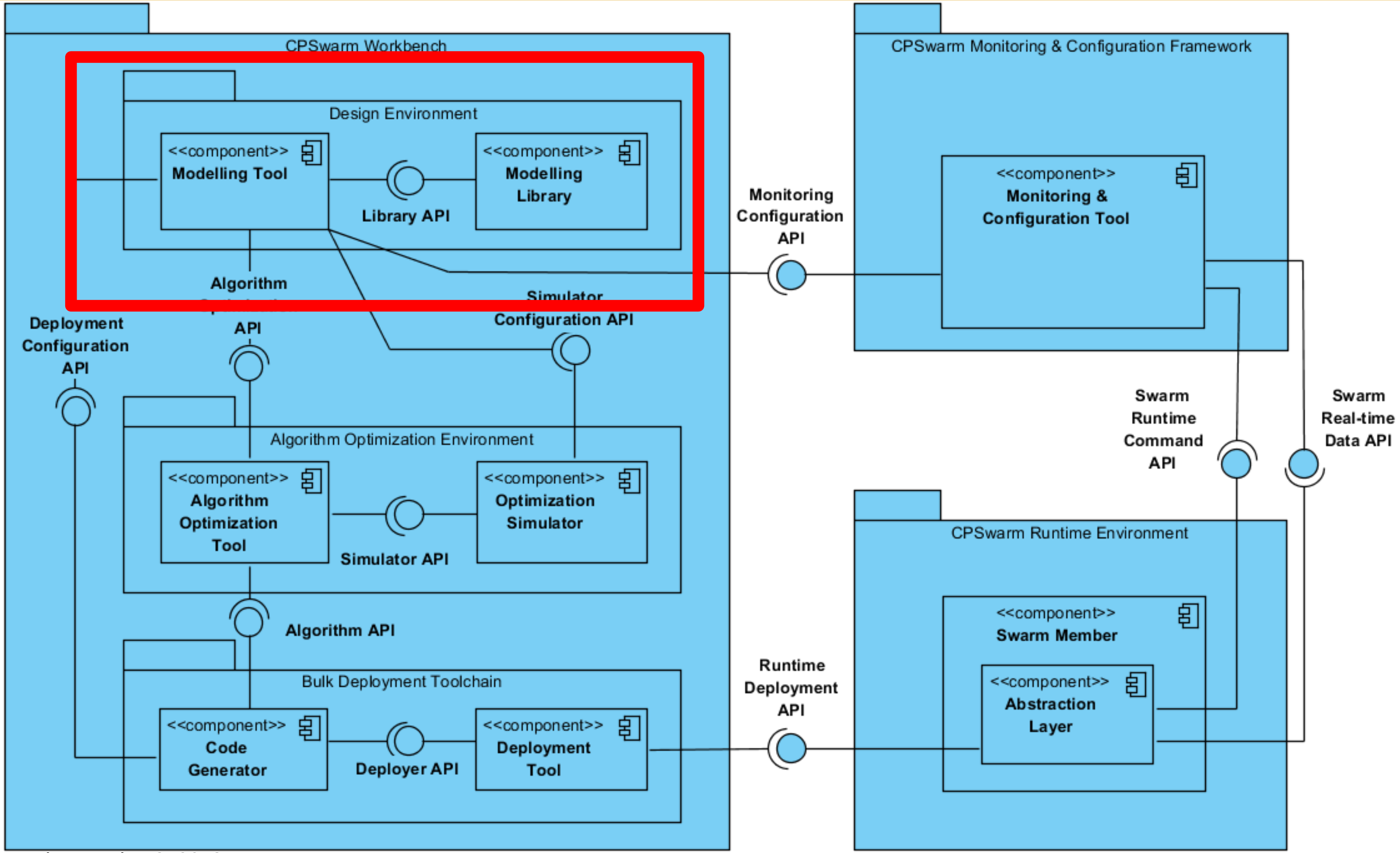
CPSwarm Stakeholders



Stakeholder	Description
Workbench Engineer	*A person, group or an organization responsible for the development and maintenance of the workbench*
Mission Planner	*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*
Swarm Designer	*A person responsible for designing the structure and behavior of the swarm based on the mission defined by the mission planner*
Domain Expert	*A person, group or an organization who is an expert of the problem domain, also in terms of rules, regulations, limitations etc.*
Security Expert	*A person, group or an organization responsible for providing expertise on safety and security of the swarm*
Swarm Modeler	*A person who constructs the structure and behavioural model of the swarm*
Algorithm Optimization and Simulation Expert	*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.*
Swarm Developer	*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.*
Deployer	*A person or group responsible for deploying the code of the swarm.*
Swarm Commander/Operator	*A person with the command control in his hand. He is responsible for directly manipulating the components of the swarm.*

WP4 Models and Algorithms for CPS Swarms

CPSwarm Architecture – high level functional view

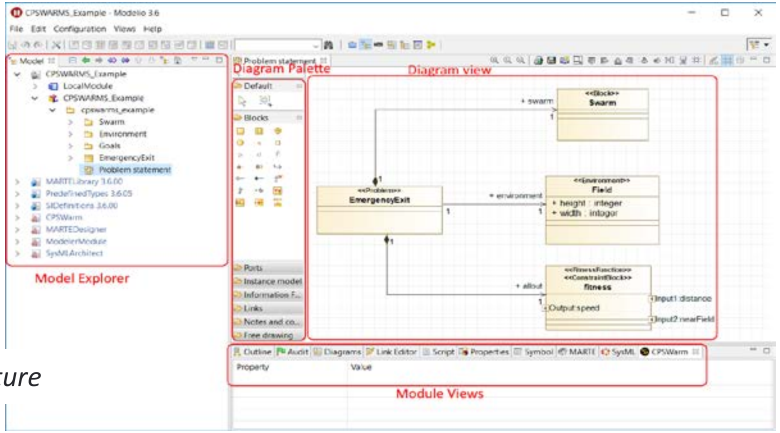


Design Environment – Modeling Tool

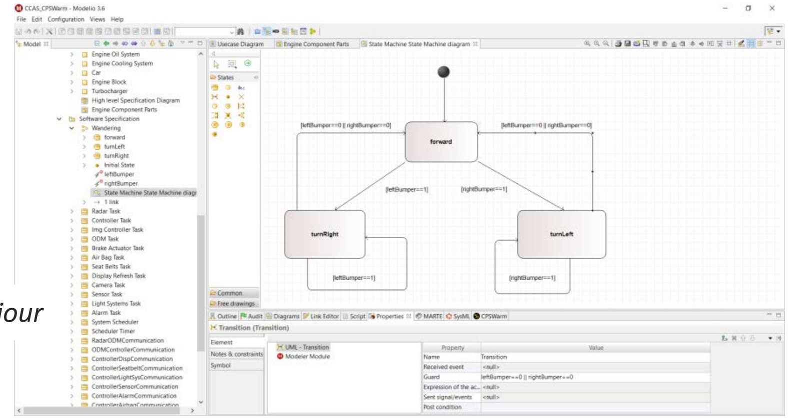
modelio
the open source modeling environment

*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

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*

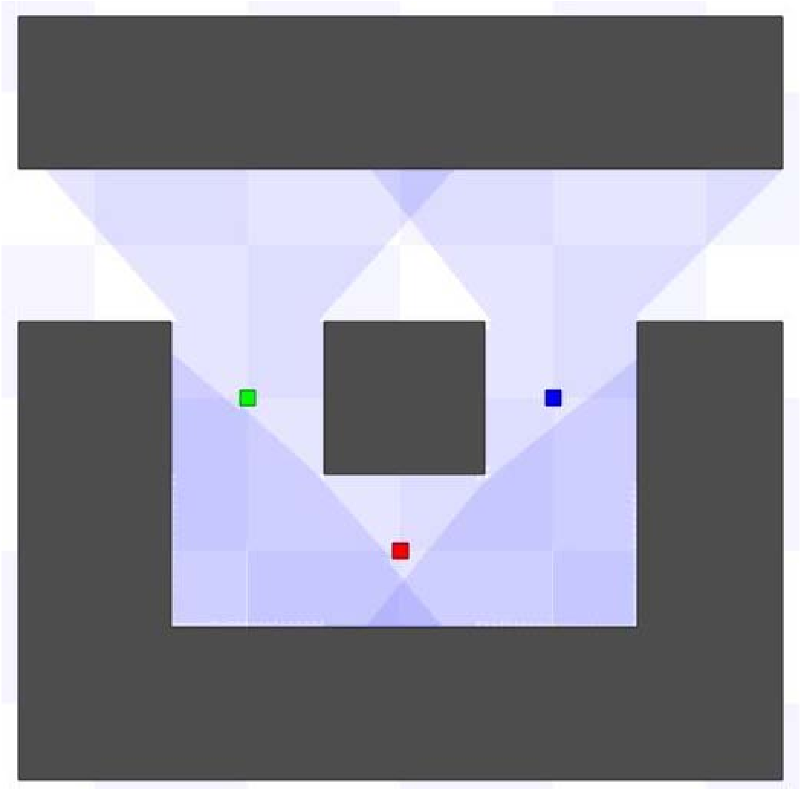
Initial CPS Modelling – on the example of the EmergencyExit

Description EmergencyExit example:

In the EmergencyExit example, multiple agents move in a 2D, discrete environment and try to find one of two emergency exits.

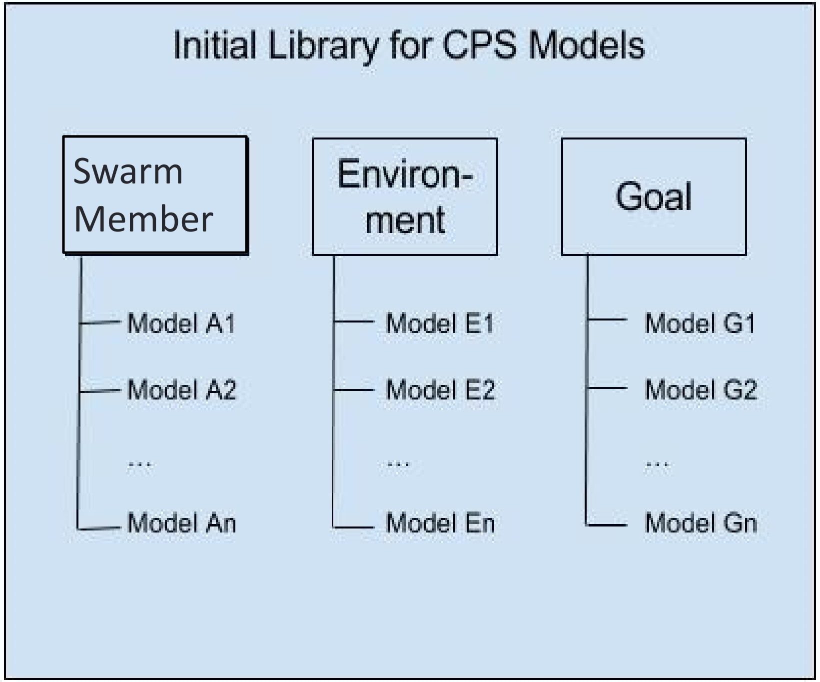
In each discrete time step, an agent senses the neighbouring cells and moves to a free cell.

When an agent reaches an emergency exit, it is removed from the environment. The goal is that all agents exit the environment.



Initial Modeling Library for CPS Models

- Overall Idea
  - Library with pre-defined models
  - Models: reused, changed, added
- Separation into three initial groups (see Figure)
  - Swarm Member
  - Environment
  - Goal
- Mandatory parts for each model in SysML
  - Unique name
  - Description
  - Parameters
    - Property: type [range]
    - Input: type [range]
    - Output: type [range]



Categories of modelling libraries

Swarm Member 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

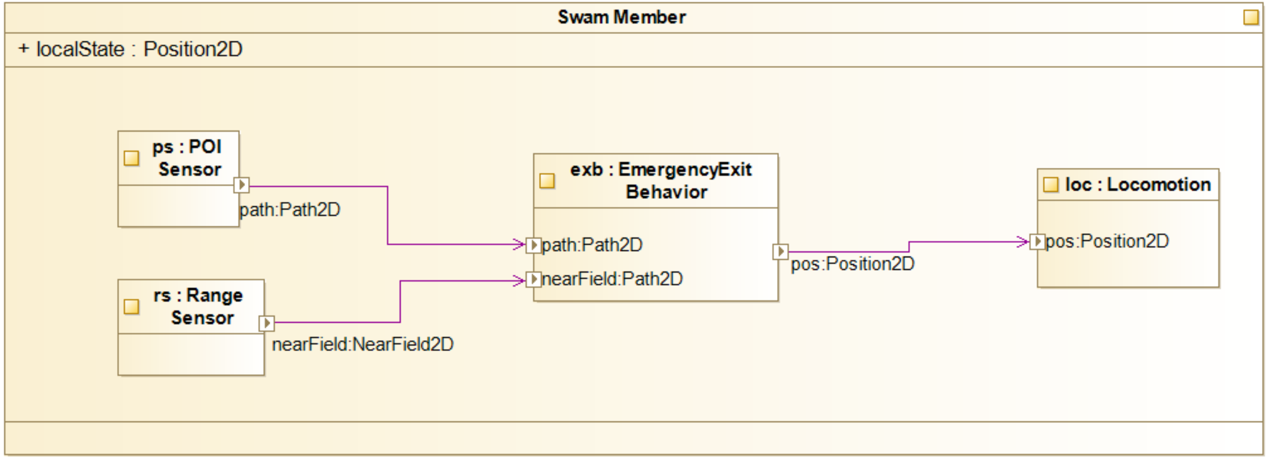
- **2D/3D map** of the environment (occupancy grid map, i.e. free space and obstacles expressed as a bitmap file)
- **Size** of the environment (width and height expressed in number of grid cells)
- **Resolution** (expressed in number of grid cells per meter)

Goal Library

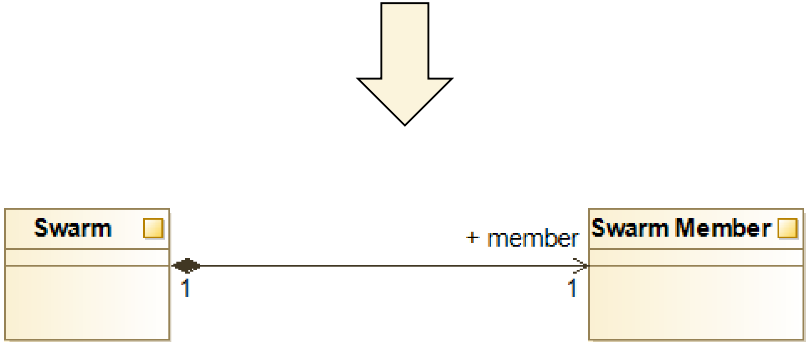
- One or multiple **fitness values**
- **Calculation specification**, actually incorporating parameters from different models

Swarm Member

- Describes a single CPS
- Sub-libraries:
  - local memory: local status, e.g. the current x/y position, available energy, etc.
  - behaviour: collecting data from sensor, performing calculations, sending data to actuators
  - physical aspects: sensors and actuators
  - security (optional)
  - human interaction (optio

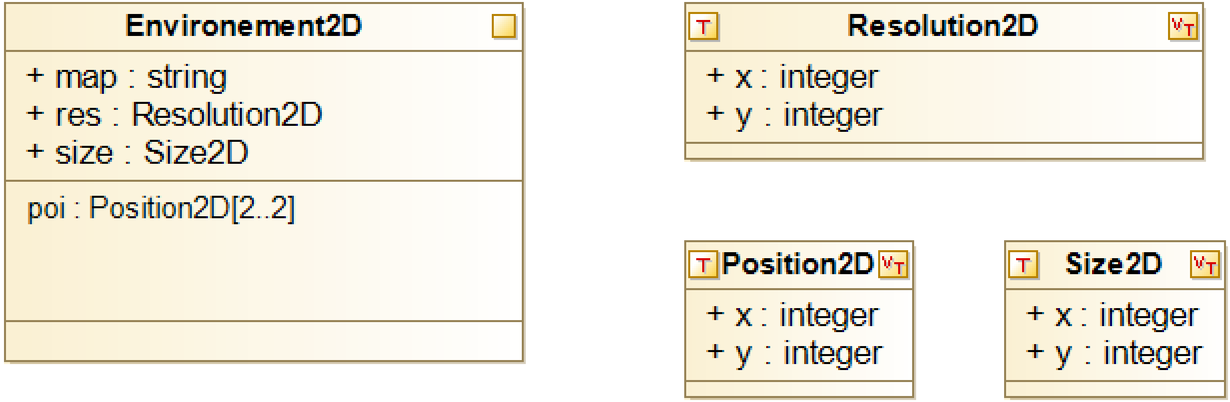


How to model a swarm



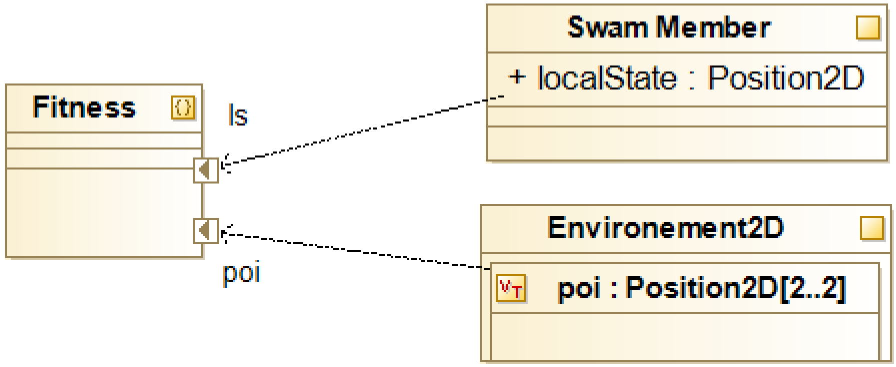
Environment (general def.)

- Describes the environment of the CPS
- Following models are mandatory, further ones can be added:
  - 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 gric
  - Resolution
    - expressed in number of gric



Goal (general def.)

- Description of the goal
- … in terms of modelling the fitness by
  - Incorporating parameters from other models
  - Calculation specification right in the model
- Multiple and multi-dimensional fitness values can be modelled



Swarm Intelligence Models and Algorithms

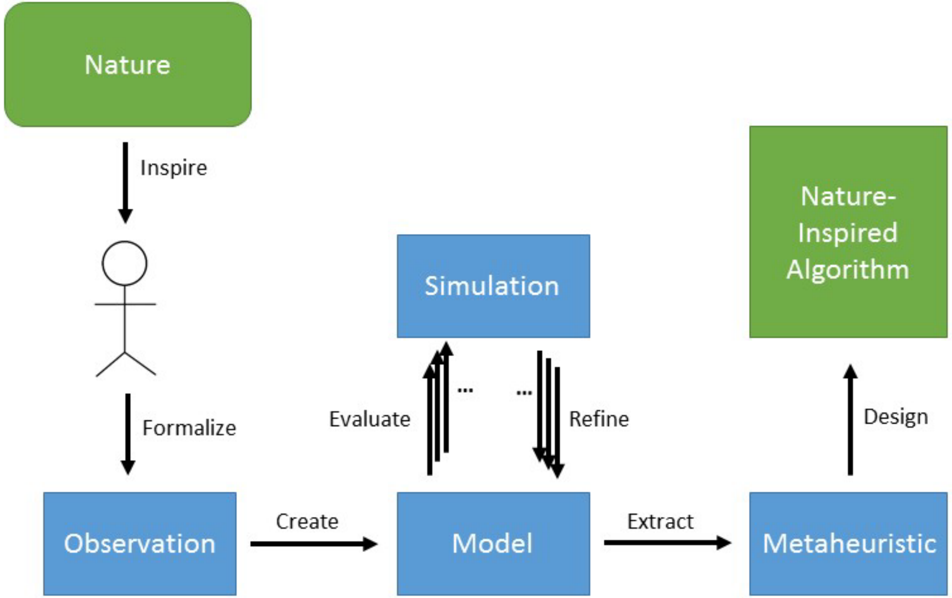
Swarm Intelligence Models

→ Process of adopting models found in nature:

- o ants, bees, fire flies, fish, etc.

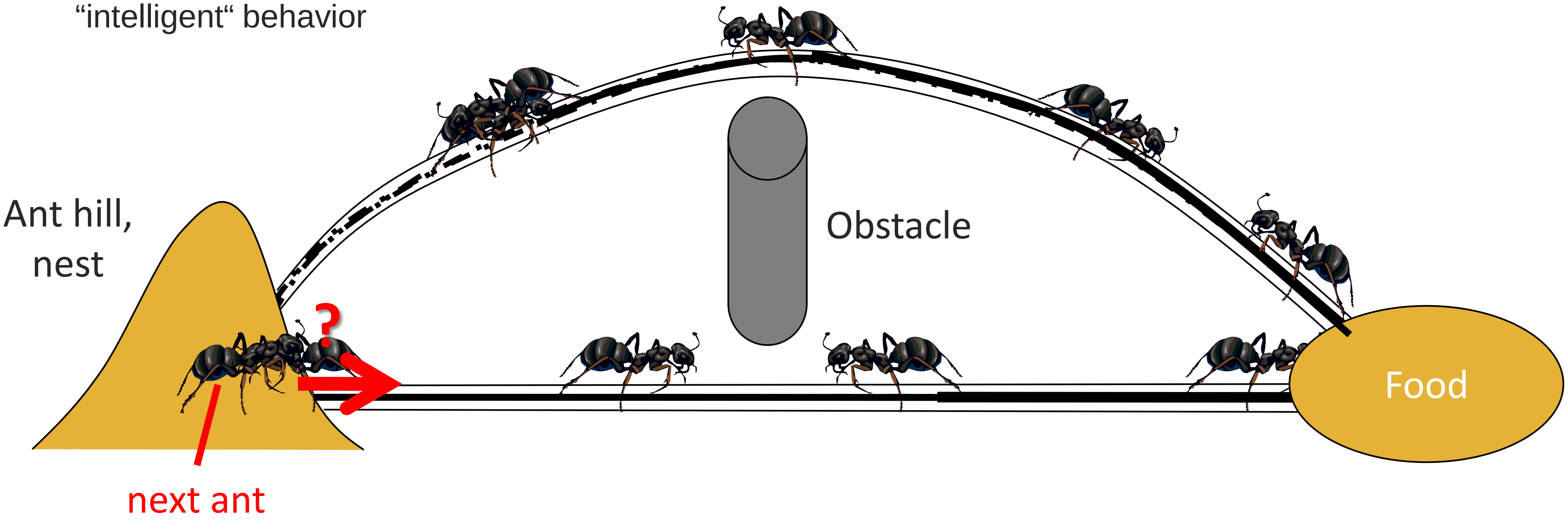
- Characteristics
  - Emergent behavior arises from simple interactions among individuals in a swarm
  - Individuals act according to simple and local behavior
  - Organized behavior emerges automatically
  - There is no central control

! NO common modeling approach **!**



Ant Routing - Inspiration

- Foraging behavior of ants
- Single ants are foolish – whole system exhibits “intelligent“ behavior



Common Modeling for Swarm Algorithms

- Part of the library Swarm Member → sub-library Behavior
- Concept adapted from the initial modelling library
  - Library with pre-defined models
  - Models: reused, changed, added
- Mandatory parts for each model
  - Unique name
  - Description
  - Parameters
    - Property: type [range]
    - Input: type [range]
    - Output: type [range]
- Degree of abstraction
  - High-level view
  - Low-level view

Modeling on the example of BEECLUST

Swarm algorithm inspired by bees, following 3 simple rules:

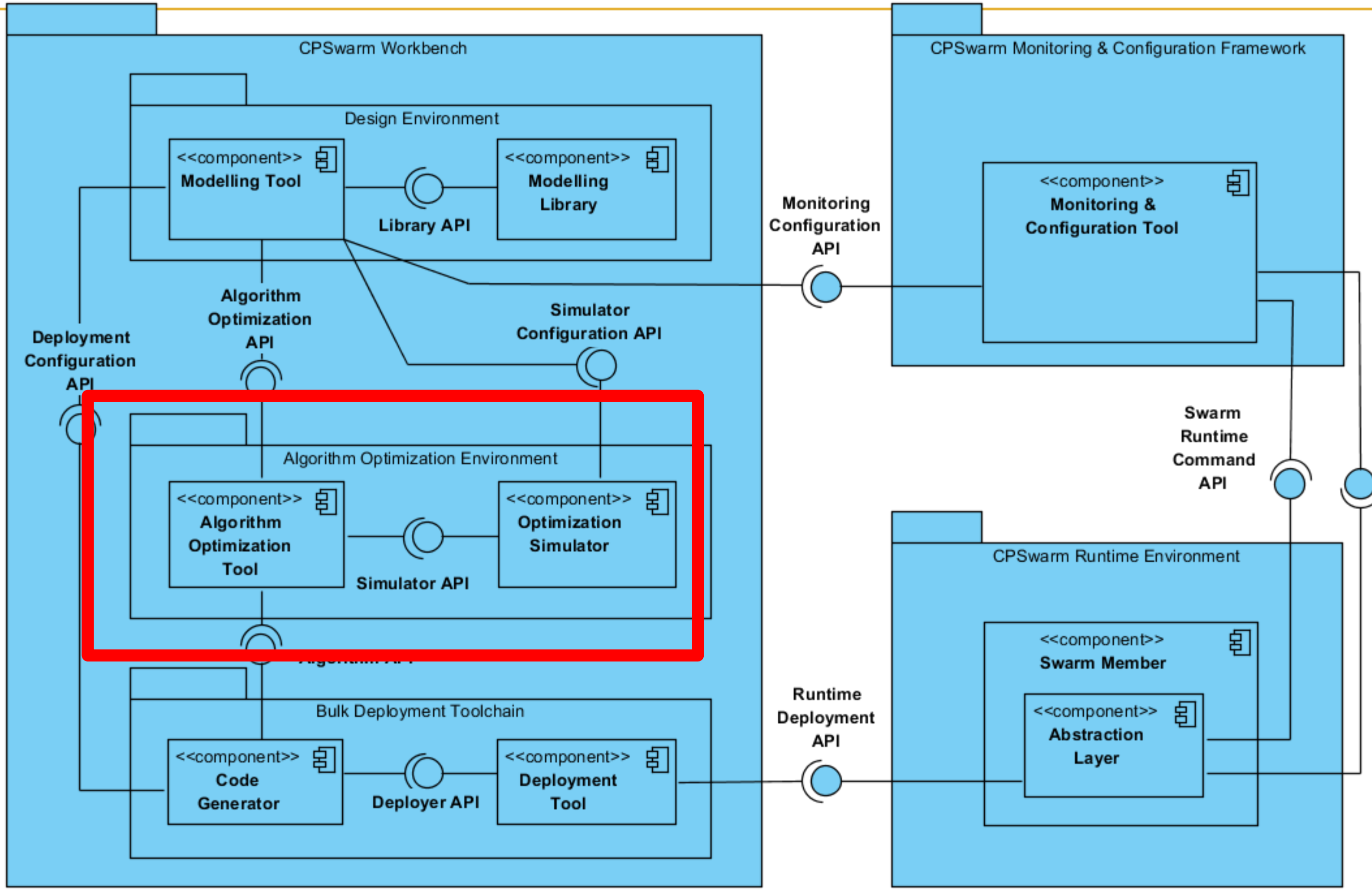
1) Move randomly
2) If a bee meets another bee: stop with waiting time w_calc
3) If a bee hits a wall: stop with waiting time w_0

Advantages for CPS:

- No direct communication among CPSs
- No indirect communication between CPS and infrastructure (stigmergy)
- No memory

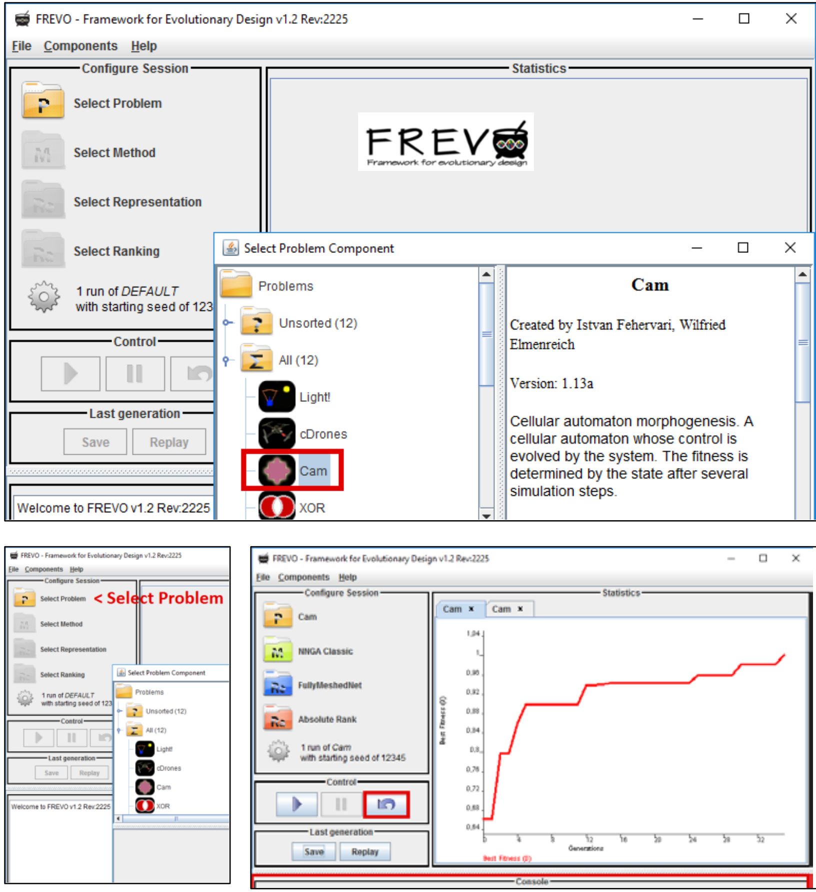
Algorithm Optimization Environment

CPSwarm Architecture – high level functional view

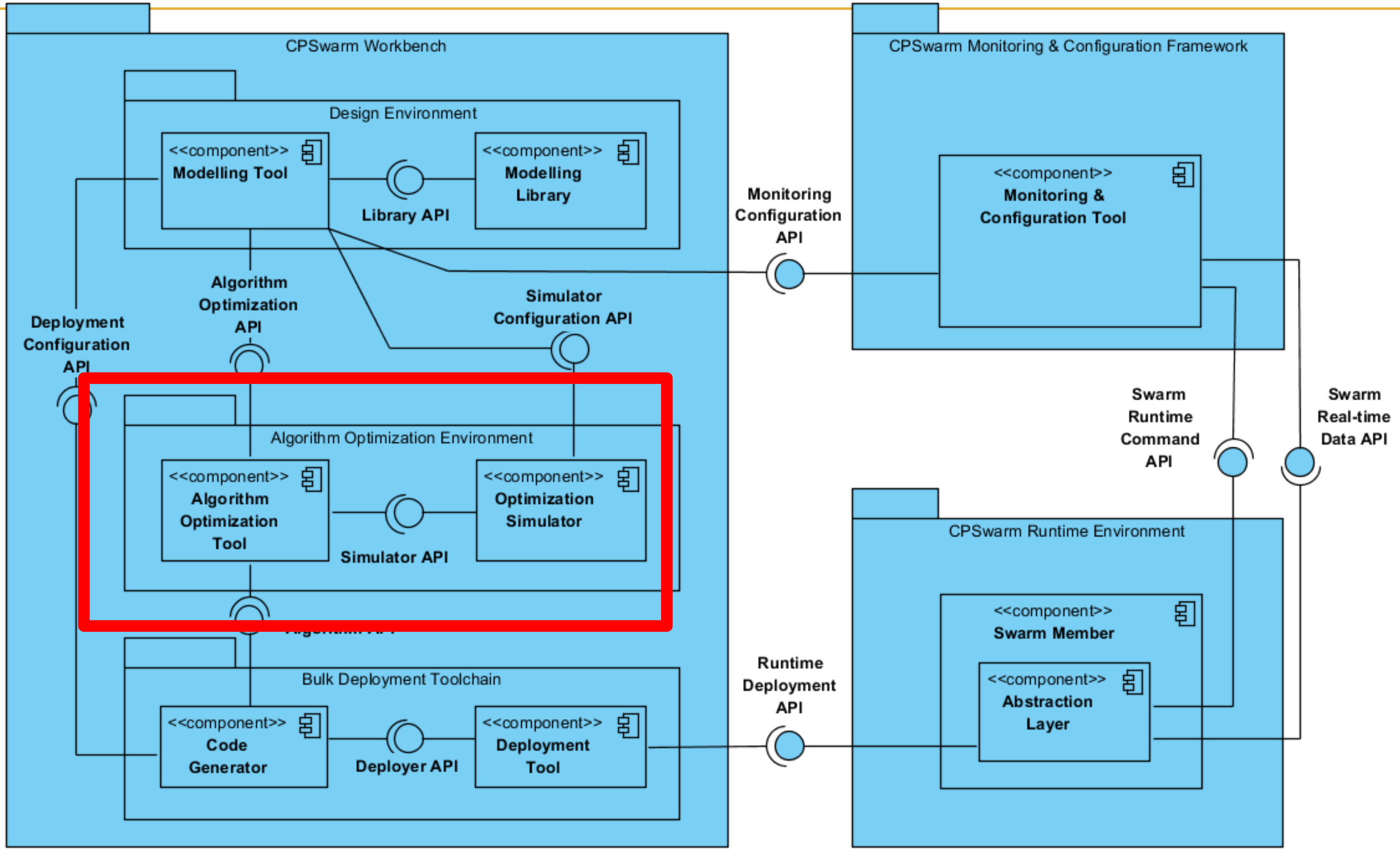


Algorithm Optimization Environment – Optimization Tool

- 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***.



CPSwarm Architecture – high level functional view



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

Live – Tutorial: FREVO

Download:

- Eclipse Neon 4.6.2
- Check Java version 1.6
- Frevo: https://sourceforge.net/projects/frevo/files/ → FREVO main packages → Frevo_v1.2.zip
- Import to Eclipse

FREVO

Framework for evolutionary design

FREVO – Installation

- Java environment (version min. 1.6)
- Eclipse (Neon 4.6.2)
- FREVO: https://sourceforge.net/projects/frevo/

Main Milestones

January 2017
Project kick-off

June 2017
Start of Use Cases Implementation

November 2017
First release of the CPSwarm workbench

November 2018
Intermediate release of the CPSwarm workbench

November 2019
Final release of the CPSwarm Workbench

December 2019
Open Source Release of Selected Software Components

December 2019
End of Use Cases Implementation

THANKS!
ANY QUESTIONs?

CONTACT

Melanie Schranz

schranz@lakeside-labs.com

lakeside-labs.com

+43 463 28 70 44/22

melanie_schranz

Coordinator

Istituto Superiore Mario Boella

Partners

Fraunhofer FIT

Lakeside Labs

TTTech

Alpen-Adria Universität

SOFTEAM Cadextan

DigiSky

SEARCH-LAB

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