Modelling CPS swarm

M. Schranz, M. Sende, A. Bagnato and E. Brosse
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.*
**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

CPSwarm Workbench

Models library
- CPS SubSystem
- CPS Functions
- Security
- Behavior
- Swarm / Self-Organization
- Human Interaction

Simulation Control and Design Optimization
- Single CPS Design
- CPS Population Design

Code Generation

Deploy Toolchain
- Bulk Deployment
- Monitoring and Configuration tools

Fitness
- Fitness Computation and Performance Comparison
- Simulation Performance Prediction

Goals

Simulators Integration Layer
- FREV
- OMNeT++
- ARGoS
- GAZEBO

CPSoS description
Simulated performance

Simulation environment

Broader Middleware e.g., IoT middleware

Data

Self-organizing Autonomous Cyber Physical System of Systems

Humans in the loop
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 uses three reference Application Scenarios to drive requirements for CPS to be deployed with developed tools

- The **Swarm Drones** scenario
- The **Automotive CPS** scenario
- The **Swarm Logistics** scenario
Swarm Modelling

How to model a swarm

Diagram showing a swarm connected to a drone with a label indicating the number of members.
Swarm Member
Communication Modeling - The basics

• An abstract model for the different communication facilities needed by the swarm was defined:
  • **Events** – to trigger state changes and exchange messages
  • **Parameters** – to establish the boundary conditions of the mission and fine tune behavior
  • **Telemetry** – to monitor the current state of swarm members
Communication Modeling - Events

- Events are the most important facility swarm members use to interact with each other
  - The state machines running on individual swarm members propagate information and trigger remote state changes using events
  - Operators issue commands by manually triggering events
- Internally, an event consists of:
  - Event name
  - List of parameters
- Events can be targeted (sent to one swarm member) or broadcasted (sent to all swarm members)
Communication Modeling - Implementation

- Communication library swarmio
  - ZMQ protocol using the Zyre library
  - Provides a C++ API for swarm members
  - Services: discovery, key-value store, event handling, pinging
  - Open: Encryption
  - Integration via swarmros: publisher/subscriber principle
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]
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.
SAR – mission summary

At the beginning of the mission:
- The drones swarm out, patrol the selected area and search for victims.
- The rovers wait for a call for intervention.

When a drone discovers one of the victims (manually controlled rover), the drone:
- communicates the position of the victim to the rovers
- starts to follow the victim and communicates possible changes of its position
- asks for further support by other drones

The rovers then
- decide which one is more suitable to reach the casualty (e.g. the closest rover is selected).
- The selected rover reaches the victim (using its GPS position received by the drones).

After the rover reaches the victim, it
- uses an emergency exit strategy to guide the victim to the closest exit.
State machine approach

- Sequence of behaviors can be extracted from the mission description
State machine approach – general definition

• Define high-level events

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Data</th>
<th>Sender</th>
<th>Scope</th>
</tr>
</thead>
<tbody>
<tr>
<td>missionStart</td>
<td>-</td>
<td>Monitoring tool</td>
<td>swarm command</td>
</tr>
<tr>
<td>missionAbort</td>
<td>-</td>
<td>Monitoring tool</td>
<td>swarm command</td>
</tr>
<tr>
<td>targetFound</td>
<td>Target ID, x/y-coordinates</td>
<td>Swarm member</td>
<td>swarm</td>
</tr>
<tr>
<td>targetLost</td>
<td>Target ID, x/y-coordinates</td>
<td>Swarm member</td>
<td>swarm</td>
</tr>
<tr>
<td>targetRescued</td>
<td>Target ID</td>
<td>Swarm member</td>
<td>swarm</td>
</tr>
<tr>
<td>emergencyEvent</td>
<td>-</td>
<td>Swarm member</td>
<td>device</td>
</tr>
</tbody>
</table>

• Define state types

High Level Behaviour: Low Level Behaviour

Abstraction Library

Do/function()
State machine approach – Drone behavior
CPSWarm Architecture
THANKS!
ANY QUESTIONS?
CONTACTS

Alessandra Bagnato
Etienne Brosse
Softeam R&D
Alessandra.bagnato@softeam.fr
Etienne.brosse@softeam.fr
@alebagnato

www.softeam.fr

http://www.cpswarm.eu

Follow @CPSwarm_EU