D5.1 - CPSWARM MODELLING LANGUAGE SPECIFICATION

Deliverable ID	**D5.1**
Deliverable Title	**CPSwarm Modelling Language Specification**
Work Package	**WP5 – CPSwarm Design Workbench**
Dissemination Level	**PUBLIC**
Version	**1.0**
Date	**07-02-2018**
Status	**Final**
Lead Editor	**Alessandra Bagnato (SOFTEAM)**
Main Contributors	**Etienne Brosse (SOFTEAM)**

Published by the CPSwarm Consortium

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

Document History

Version	Date	Author(s)	Description
0.1	2017-11-13	Etienne Brosse (SOFTEAM)	Table of Content
0.2	2017-12-08	Etienne Brosse (SOFTEAM)	First draft
0.3	2017-12-08	Alessandra Bagnato (SOFTEAM)	Consolidated version
0.4	2017-12-21	Etienne Brosse (SOFTEAM)	Integrate LAKE comments
0.5	2017-12-22	Etienne Brosse (SOFTEAM)	Integrate KLU comments
0.6	2018-01-08	Etienne Brosse (SOFTEAM)	Integrate ISMB comments
0.7	2018-01-09	Alessandra Bagnato (SOFTEAM)	Consolidated version, dynamic aspects addressed
1.0	2018-02-07	Alessandra Bagnato (SOFTEAM)	Minor modifications inserted. Document ready to be submitted to EC

Internal Review History

Review Date	Reviewer	Summary of Comments
2017-12-20	LAKE	Typos and needs of more complete example
2017-12-21	UNIKLU	Typos

Contents

1 Executive summary

This deliverable, namely "**D5.1 - CPSwarm Modelling Language Specification**", introduces the results of CPSwarm **Task 5.1 - Design Environment Language** with the project language used for CPS Swarm modelling.

T5.1 ambition was to develop a dedicated CPSwarm language on top of SysML. The Systems Modeling Language (SysML) is a general-purpose visual modeling language for systems engineering applications. SysML is defined as a dialect of the Unified Modeling Language (UML) standard, and supports the specification, analysis, design, verification and validation of a broad range of systems and systems-of-systems. These systems may include hardware, software, information, processes, personnel, and facilities. With the CPSwarm CPS Swarm modelling language we used SysML concepts following the CPSwarm end user modeling needs (defined in WP2).
The goal of the task 5.1 is to define a dedicated CPSwarm UML2 Profile defining SysML customizations, and a specific graphical notation. SOFTEAM leads the task and the deliverable while ISMB and FRAUNHOFER supported the task activities with knowledge on UML and semantic models of devices and systems.

2 Introduction

2.1 Scope

This document provides an initial/working version of the modelling language definition- based on UML and SysML - in the CPSwarm context. It will be updated further during the study.
The intent of this document is to specify the diagram extension to guide users during their modelling.

2.2 Document organization

This deliverable starts by giving an overview of the CPSwarm Modelling language (Section 3).
Then Sections 4, 5, 6, 7, 8 and 9 give more details for each view proposed in the language. Finally, Section 10 concludes on the current state of CPSwarm modelling language.

2.3 Related documents

ID	Title	Reference	Version	Date
[D4.1]	Initial CPS Modelling Library	D4.1	1.0	M9
[D6.1]	Initial Simulation Environment	D6.1	1.0	M9
[D5.2]	CPSwarm Modelling Tool	D5.2	1.0	M9
[D2.1]	Initial Vision Scenarios and Use Case Definition	D2.1	1.0	M4
[D3.1]	Initial System Architecture Analysis & Design Specification	D3.1	1.0	M6

3 CPSWarm Modelling Language

One of the main goals of the CPSwarm project is to analyse the current UML/SysML specifications and to develop a CPSwarm Modelling language which will be suitable for CPS swarm and autonomous system development. Currently SysML is advertised as a language for providing model-based description of systems. However, during our initial analysis, we have found that the SysML language cannot adequately support the domain-specific needs of CPS swarm and autonomous systems. The identified shortcomings of SysML are briefly discussed below.

- First, the SysML language deals mainly with system engineering while environment and goal specifications are not explicitly defined. The CPSwarm Modelling language specification tries to fill this gap by covering these concepts.
- Second, the usage of SYSML profile often requires expert knowledge of the UML metamodel, which is unnecessary in day-to-day UML modelling. This is because of the fact that even if each SysML element definition is clear, these concepts often extend several UML metaclasses. This means the SysML stereotypes are applicable to UML elements with completely different semantics, which usually generate a frustration at the user side.

Many people – coming from multiple domains – can be involved during CPS swarm designing. A correct aspect separation allows users to work more efficiently by simplifying and formalizing modelling means and the methodology cf. [D3.1]. The following sections describe a set of diagrams supporting the CPSWARM methodology. This set is composed of customized diagrams dedicated to the methodology as well as several UML diagrams with extensions.

For each of the diagrams the following descriptions are provided:

- Overview - A brief diagram description;
- Diagram Elements - A table of elements to be used in a context;
- Example - A diagram example presenting an application scenario.

4 Problem Statement

4.1 Overview

The CPSwarm Problem Statement diagram is based on the UML Class diagram. Its main goal consists in exposing a problem and reflecting the relations existing between it and its related element i.e. a Swarm, one or many Environment(s), and a Fitness Function.

4.2 Diagram Palette

Name	Icon	Representation	Description
Problem	?	<<Problem>> Problem	The CPSwarm Problem represents what CPSwarm project i.e. defining the best CPS swarm – deployed in a predefined set of environments – according to a given goal.
Swarm		<<Swarm>> Swarm	The Swarm is a group of CPS (aka swarm members) all moving together in a predefined environment to achieve a goal**.**
Environment		<<Environment>> Environment	The Environment represents the field in which the swarm will be involved. Different kind of constraint - such as size or obstacles– can be set to test the swarm robustness against.
Fitness Function		<<FitnessFunction>> Fitness Function	The Fitness Function is defined to evaluate the modelled swarm. One or many criteria might be defined.
Usage	⇢	<<use>> - - - - - - - - - - - - - ->	Usage dependencies are used to wire a Problem to the its related Swarm, Environment, and Fitness Function.

Table 1: Problem Specification diagram elements

4.3 Examples

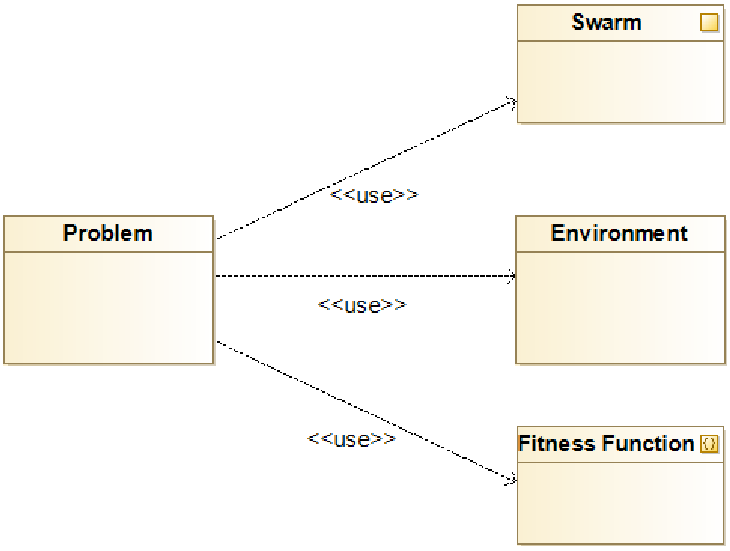


Figure 1 Problem statement diagram example

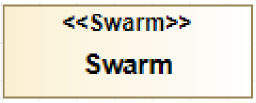
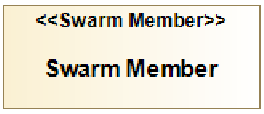
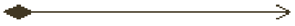
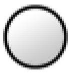
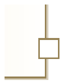
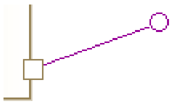
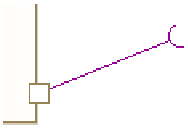
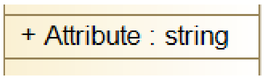
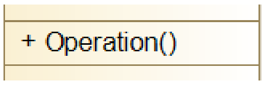
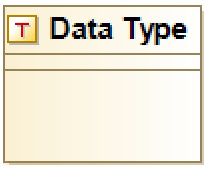
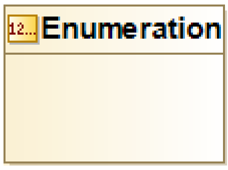
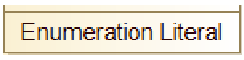
Figure 1 shows the typical example of Problem Statement. The central concept of each Problem Statement is the Problem named "Problem" here. A problem is fully defined when its related Swarm, Environment and Fitness Function are specified.

5 Swarm Architecture

5.1 Overview

Swarm Architecture diagrams are used for designing the Swarm composition in terms of Swarm Member. It also can be used to describes the external interface of a Swarm Member.

5.2 Diagram Palette

Name	Icon	Representation	Description
Swarm			The Swarm is a group of swarm members all moving together in a predefined environment to achieve a goal.
Swarm Member			A Swarm Member represents a kind of CPS (individual) involved in a given swarm.
Composition			Composite aggregation is a strong form of aggregation that requires the target (or part) object be included in at most one composite object.
Interface	○		Interfaces declare services implemented by components which implement the Interfaces.
Port			A Port is used for data flow communications between components.
Provided Interface			The Provided Interfaces of a Port expose service (Interface) provided by the port owner to its environment.
Required Interface			The Required Interfaces of a Port characterize service (Interface) needed by the port owner to its environment.
Attribute	A:		An Attribute is a typed property of its owner.
Operation	0()		Operations represent invokable behaviour.
DataType	T		DataType are Type elements used to model pure values.
Enumeration	12...		Enumeration is a kind of DataType. Each value of an Enumeration corresponds to one of its user-defined EnumerationLiterals.
EnumerationLiteral			An EnumerationLiteral is a user-defined data value for an Enumeration

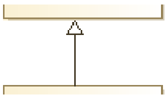
Generalization	⇧		Generalization corresponds to the usual Object Inheritance concept.

Table 2: Swarm Architecture diagram elements

5.3 Examples

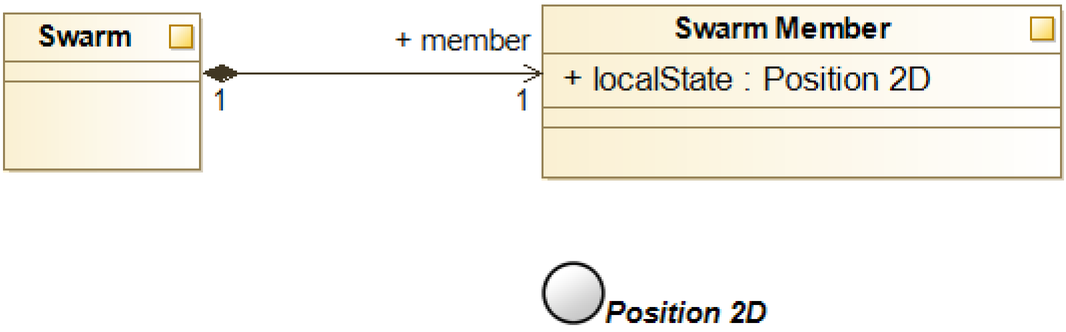


Figure 2 Swarm Architecture and Swarm Member Interface example

The Figure 2 represents the simplest possible swarm which is composed of one unique Swarm Member. Swarm Architecture diagram is also used to depict the "localState" Attribute of the Swarm Member of type Position2D.

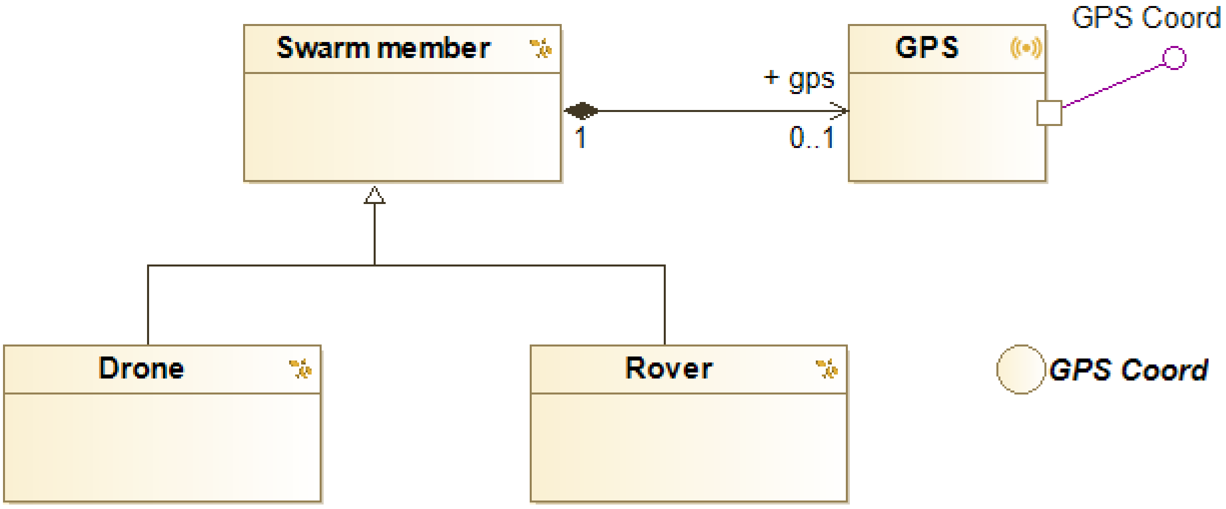


Figure 3 Heterogonous swarm example

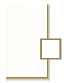
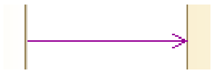
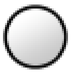
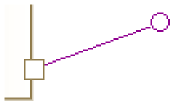
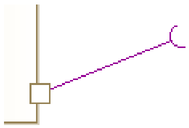
The Figure 3 depicts the fact that both "Drone" and "Rover" are possible "Swarm member ". A swarm Member might ("0..1" cardinality) have a "GPS" sensor. This "GPS" sensor provides, through a port "GPS Coord".

6 Swarm Member Architecture

6.1 Overview

Swarm Member Architecture diagrams are used for designing the internal architecture of a Swarm Member. This diagram allows users to express the various internal components and also the flows between them by using the port and the connector concepts.

6.2 Diagram Palette

Name	Icon	Representation	Description
Swarm Member		<<Swarm Member>> Swarm Member	A Swarm Member represents a kind of CPS (individual) involved in a given swarm.
Actuator	⊙	<<Actuator>> Actuator	An Actuator is a piece of CPS in charge of realizing a physical action (e.g. moving an arm).
Controller		<<Controller>> Controller	A Controller is a core component of a CPS. It contains the behaviour (or part of the behaviour) of the CPS owner.
Sensor	((•))	<<Sensor>> Sensor	A Sensor is a CPS component which analyses the CPS environment and provides data to the controller.
Part		□ s : Sensor	A Part represents the internal instantiation of a given Component e.g. a Controller, an Actuator, or a Sensor.
Port			A Port is used for data flow communications between components.
Connector	→		A Connector represents a specific interaction path between ports.
Interface	●		Interfaces declares services implemented by components, which implement the Interfaces.
Provided Interface			The Provided Interfaces of a Port expose service (Interface) provided by the port owner to its environment.
Required Interface			The Required Interfaces of a Port characterize service (Interface) needed by the port owner to its environment.
Attribute	A:	+ Attribute : string	An Attribute is a typed property of its owner.

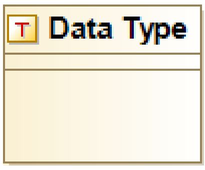
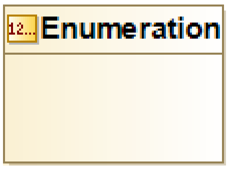
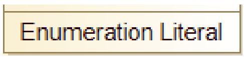
DataType	T		DataType are Type element used to model pure values.
Enumeration	12...		Enumeration is a kind of DataType. each value of an Enumeration corresponds to one of its user-defined EnumerationLiterals.
EnumerationLiteral	▸x◂		An EnumerationLiteral is a user-defined data value for an Enumeration

Table 3: Swarm Member Architecture diagram element

6.3 Examples

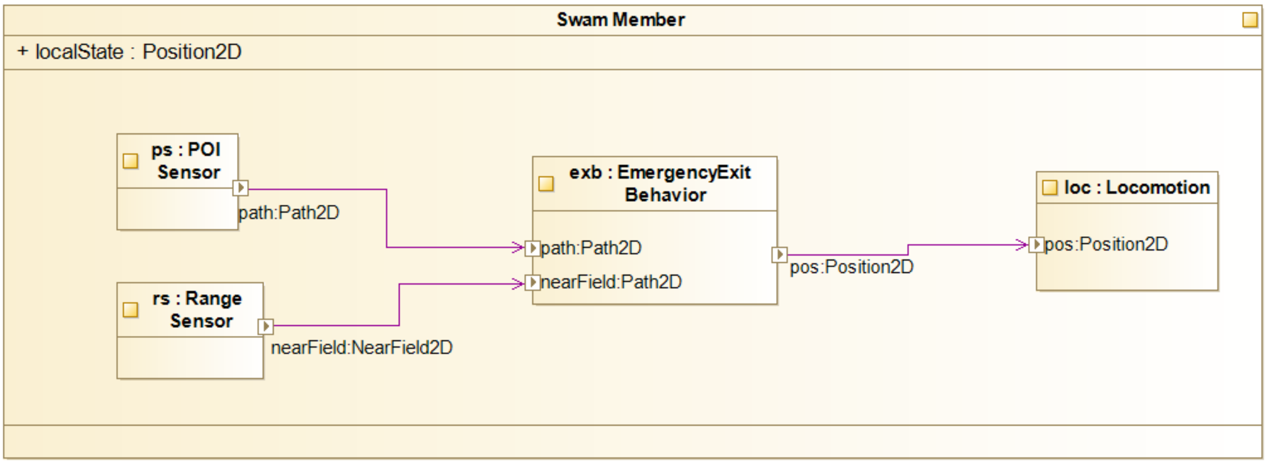


Figure 4 Swarm Member Internal Architecture Example

Figure 4 depicts the internal architecture of a Swarm Member already partially describes in Figure 2. This diagram still shows the “localState” attribute but also four internal components (parts) respectively named “ps”, “rs”, “exb”, and “loc”. These four parts are connected through specific point (port) named “path, nearfield, or “pos”. The data going trought the connection is model thanks to interface (not shown in this diagram) but named “Path2D”, “NearField2D”, and “Position2D”.

7 Environment Architecture

7.1 Overview

Environment Architecture diagrams are used to design the Environment in which the swarm is going to operate in. This diagram is based on UML class diagram and is used to generate Java code for optimization purpose.

7.2 Diagram Palette

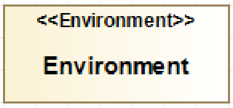
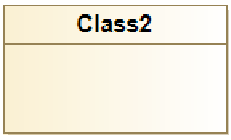
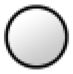
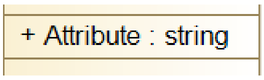
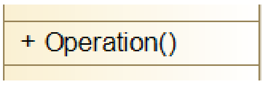
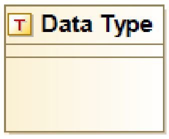
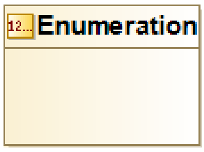
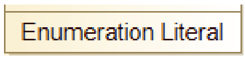
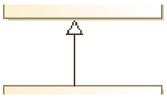
Name	Icon	Representation	Description
Environment			The Environment represents the field in which the swarm is going to operate in. Different kind of constraints - as size, or obstacles – can be set to test the swarm robustness in dynamic environments.
Class			A Class represents any abject related or deployed in the Environment.
Association	↦		An Association describes discrete links among two classes.
Aggregation	◇—		Aggregation is a "weak" form of composition.
Composition	◆—		Composite aggregation is a strong form of aggregation that requires a part object be included in at most one composite object.
Interface	○		Interfaces declare services implemented by components which implement the Interfaces.
Attribute	A:		An Attribute is a typed property of its owner.
Operation	0()		Operations represent invokable behaviour.
DataType	T		DataType are elements used to model pure values and not Object.
Enumeration	12...		Enumeration is a kind of DataType. Each value of an Enumeration corresponds to one of its user-defined EnumerationLiterals.
EnumerationLiteral			An EnumerationLiteral is a user-defined data value for an Enumeration
Generalization	⇧		Generalization corresponds to the usual Object Inheritance concept.

Table 4: Swarm Architecture diagram elements

7.3 Examples

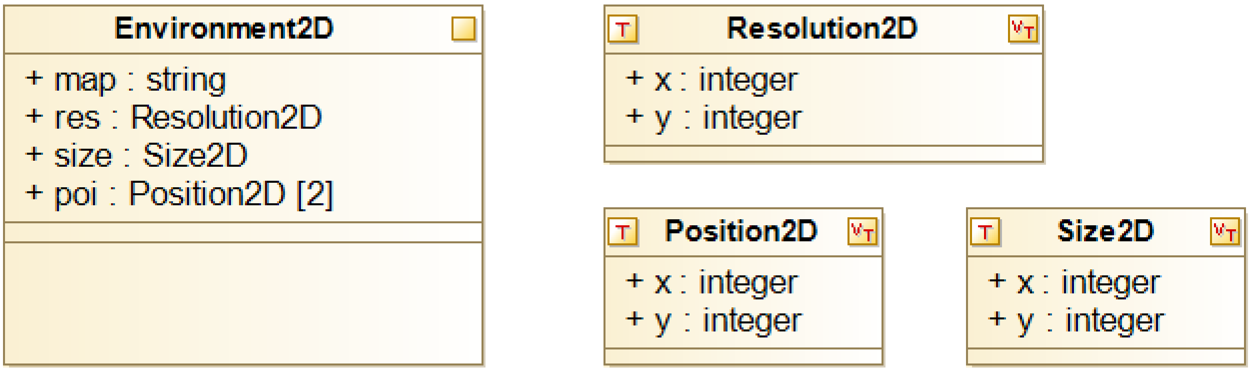


Figure 5 Environment Example

Figure 5 represents a 2D environment named “Environment2D”. This Environment is described by a map (bmp file in the computer), a 2D Resolution “res”, a 2D Size “size”, and two points of interest “poi” with their 2D position, respectively.

8 Behavioural Modelling

8.1 Overview

In CPSwarm methodology, State machine diagrams are used to describe the dynamical (across time) aspect of all objects. This dynamical aspect can represent the behaviour of a CPS swarm, a single CPS, an Object in the environment, etc. State machine diagrams describe a set of the possible states that a specific object may be in and the transitions between those states. A state represents a stage in the behaviour pattern of an object.

8.2 Diagram Palette

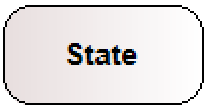
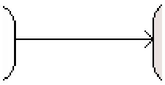
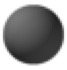
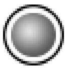
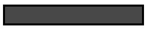
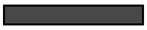
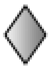
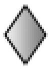
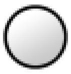
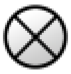
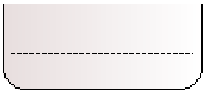
Name	Icon	Representation	Description
State	▭		State models a notable situation during the life of an object.
Transition	}→{		Transition is a directed relationship between two states.
Initial PseudoState	●		Initial PseudoState represents a default state - the starting state- that is the source for a single Transition to the default state of a composite sate
Final PseudoState	◉		Final PseudoState signifies that the enclosing region is completed.
Fork PseudoState	⇉		A Fork PseudoState is a PseudoState that splits an incoming Transition into two or more outgoing Transitions.
Join PseudoState	⇉		Join PseudoState is used to merge several transitions emanating from source in different orthogonal regions.
Merge PseudoState	▷		A Merge PseudoState is a PseudoState that merges conditional branches.
Choice PseudoState	◇		A Choice PseudoState is a PseudoState that realizes a dynamic conditional branch.
Entry PseudoState	○		An Entry PseudoState is an entry point of a state machine or a composite state.
Exit PseudoState	⊗		An Exit PseudoState is an exit point of a state machine or a composite state.
Region	▭		Region is an orthogonal part of either a composite state or a state machine.

Table 5: Swarm Member Behaviour diagram element

8.3 Examples

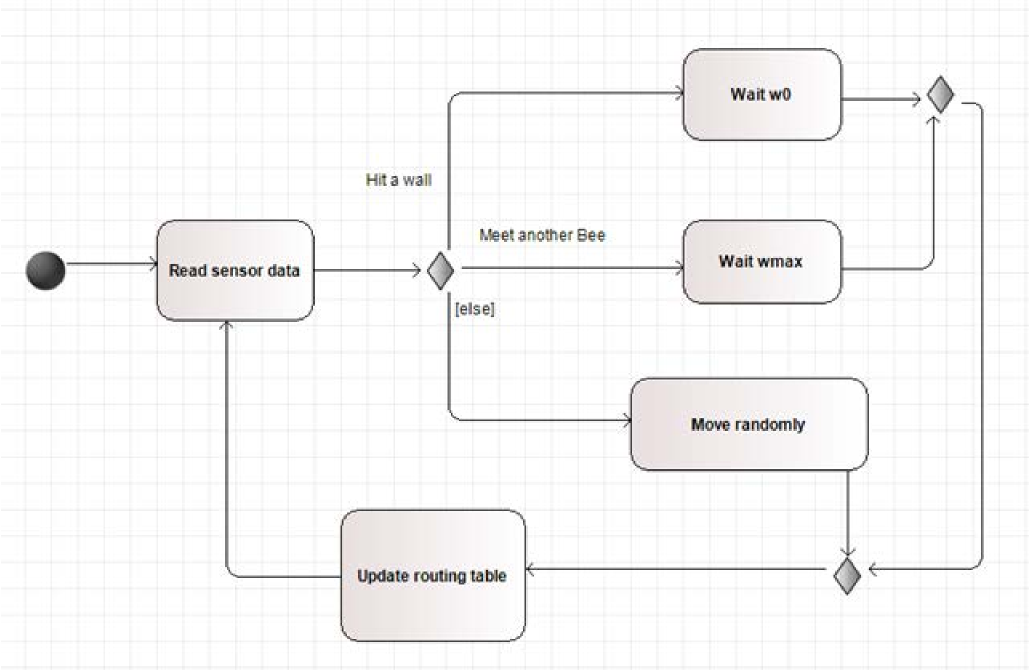


Figure 6 Swarm Member Behaviour Example

The Swarm Member behaviour is shown in Figure 6 designed with five states “Read sensor data”, “Update routing table”, “Wait w0”, “Wait wmax”, and “Move randomly”. The first state reached is the “Read sensor data”. According to the data collected the Swarm Member waits for a specified time w0, for another one, called wmax, or moves randomly. In any case, he updates its routing table and then reads again sensor data.

9 Fitness Function Specification

9.1 Overview

The CPSwarm Function Fitness Function Specification diagram is based on the SysML Parametric diagram. Its main goal consists in exposing CPS swarm goal, sub-goal, and reflecting the relations existing between them. It specifies the capability or condition that should be optimized (minimized or maximized) by the problem definition (swarm and environment).

9.2 Diagram Palette

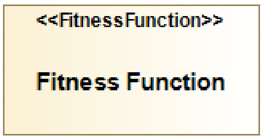
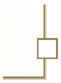
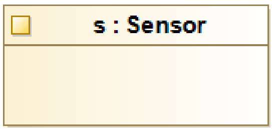
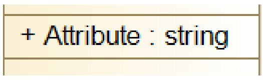
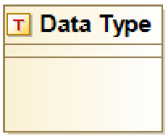
Name	Icon	Representation	Description
Fitness Function			The Fitness Function is defined to evaluate the modelled swarm. One or many criteria might be defined.
Port			A Port is used for data flow communications between components.
Part			A Part represents the internal instantiation of a given Component e.g. a Controller, an Actuator, or a Sensor.
Attribute	A:		An Attribute is a typed property of its owner.
DataType	T		DataType are element used to model pure values and not Object.
Reference	-->		Reference is used to point from value defined in the CPS Swarm architecture definition to the entries (port) of the Fitness Function

Table 6: Fitness Function diagram element

9.3 Examples

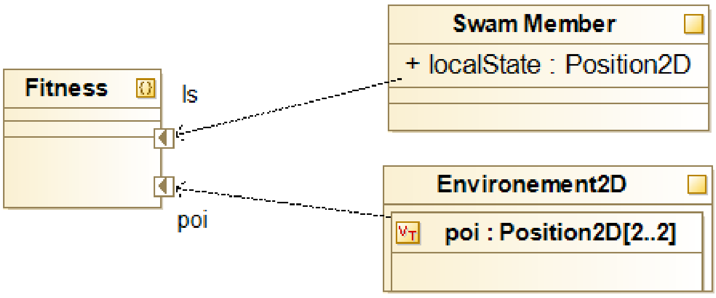


Figure 7 Fitness Function example

Figure 7 shows a Fitness Function “Fitness” which used two values named “ls” and “poi”. These values, respectively, come from the “localState” of the “Swarm Member – shown in Figure 4 and Figure 6 - and the point of interest “poi” defined in Figure 5.

10 Conclusion

This deliverable describes the status of CPSwarm modelling language. It illustrates graphical rendering of the CPSwarm modelling based on the profile definition with several examples. Each of these example is dedicated to a specific aspect of CPS warm problem definition.

The work presented here paves the way to a complete and useful Model-based design methodology for the specification of swarm of Cyber-physical system.

In the following years of CPSwarm project, we expect:

- Redefinition(s) of the diagrams presented here to feet end user needs concerning swarm problem modelling.
- Introduction of mathematical expressions for goal/fitness function specification
- Addition of new diagrams to model other aspect of CPS swarm as human in the loop aspect for example.

In D5.3 we will take into account the dynamic environment aspect that will emerge in the complex demos scenarios to be prepared for the next reporting period and review. Currently the best way to model this aspect is under study. State machines are at the moment of writing the chosen solution.

D5.3 will also address safety, security. Those aspects will be specified in D5.3 and then traced in the toolchain promoted architecture

Acronyms

Acronym	Explanation
SysML	System Modeling Language
UML	Unified Modeling Language
CPS	Cyber Physical System

List of figures

List of tables