Department of Distributed and Dependable Systems
Technical report no. D3S-TR-2019-02
December 15, 2019

Department of Distributed and Dependable Systems D3S

Uncertainty-Aware Self-Adaptive Component Design in Cyber-Physical System

Rima Al-Ali

Abstract: Cyber-physical systems (CPS) need to be designed to deal with various forms of uncertainty associated with data contributing to the system's knowledge of the environment. Dealing with uncertainty requires adopting an appropriate model which then allows making the right decisions and carrying out the right actions (possibly affecting the environment) based on imperfect information. However, choosing and incorporating a suitable model into CPS design is difficult, especially for non-experts, because it requires identifying the kind of uncertainty at hand as well as knowledge of suitable models and their application to dealing with the uncertainty. While inspiration can be found in other CPS designs, the details of dealing with uncertainty in another CPS can be confounded by domain-specific terminology, context, and requirements. To make this aspect of CPS design less daunting for non-experts, we aim at providing an overview of approaches dealing with uncertainty in the design of CPS targeting collective behavior. To this end, we present a systematic review of relevant scientific projects with industrial leadership and a synthesis of relations between system features, the kinds of uncertainty, and models and methods used to deal with it. The results provide an overview of uncertainty across different domains and challenges and reason about a guide for designing uncertainty-aware self-adaptive components in CPS.

1 Introduction

Cyber-physical systems (CPS) [22] are an emergent trend in everyday lives. Essential properties of CPS include the ability to observe its environment or monitor an associated physical process, the ability to perform computations to reach decisions, and the ability to influence its environment or the associated physical processes. While the physical part of a CPS naturally deals with continuous time, the computational part deals with discrete time, leading to discrepancies between actual and perceived system state. Combined with sensing inaccuracy, delays and loss of data in communication, and delays and inaccuracy in actuation give rise to significant amount of uncertainty [38] that needs to be considered by the decision-making part of the system. In adaptive systems, these issues have a significant impact on the adaptation decisions [11, 1, 37]. This makes designing and developing a CPS difficult.

When designing a CPS, a software developer needs to: (1) understand the domain context and requirements with respect to dealing with uncertainty in CPS, (2) determine the type of uncertainties that need to be dealt with, (3) identify an appropriate model for the uncertainty and incorporate it in the design, and (4) assess the impact of uncertainty and the adopted models on the CPS requirements and runtime decisions. This can be a demanding and error-prone task for any developer, especially a non-expert one.

To provide a particular example, consider the following situation: The choice of models [29] is related to controllable and uncontrollable design variables, constraints, and responses. The Taguchi approach is not applicable at runtime and does not provide information about the influence of uncontrollable factors [10]. There are no clearly defined relations between choice of models and methods for dealing with uncertainty and higher-level system features and requirements. The missing information can then lead to software design problems such as inconsistent reliability levels [28].

To aid non-expert developers to deal with uncertainty during the CPS design, our goal is to develop a methodology that simplifies the task of identifying the types of uncertainty in a CPS, as well as identifying an appropriate corresponding model or method to be incorporated into the CPS software design.

To discover relationships among CPS requirements, various types of uncertainty, models, and methods, we conducted a systematic study of research projects related to collaborative CPSs. Because gaining access to internal industrial research projects is unrealistic, we focus on large EU projects under industrial leadership, involving partners from both academia and industry. The presence of industrial partners ensures that the projects deal with topics which the industry considers innovative and relevant for the future, while the presence of academic partners ensures that the projects will be scientifically relevant. In addition, such projects often include practical solutions and demonstrators next to the deliverables.

The study follows the guidelines for performing systematic literature reviews (SLR) in software engineering [18]. However, instead of applying the approach to scientific papers, we apply it to EU project descriptions and deliverables. This is because unlike scientific papers, industrial projects provide demonstrators, as well as more complete description of requirements and challenges to be addressed.

We present the results of the study along with a synthesis of dependencies between factors in the system, showing relationships between system features and modeling decisions. This helps non-experts to discover the kinds of uncertainty they are dealing with during CPS design, along with typical models and methods used for handling the uncertainty, depending on the domain and challenges being addressed. While some studies have related uncertainty to other aspects of a system, such as self-adaptation [27], a specific challenge, or a domain such as bio-fuel supply chain [4], the existing literature does not provide an overview of the types of uncertainty across domains in industrial use cases.

We start with introducing an overview on the background of the study and a brief summary of the guidelines.

1.1 Background

Developing devices that uses software and hardware parts to perform a specific task resulted in having Embedded Systems. However, in some cases, the device needs to regulate and adapt its own behavior, which usually requires adding a feedback loop. This loop connects its physical and cyber parts forming Cyber-Physical Systems. Nowadays, due to the revolution of Industry 4.0 and the growth of using Internet-of-Things (IoT), there is a need for cooperative behavior between different devices, which led to introducing a new kind of systems called smart Cyber-Physical Systems (sCPS). Our study focuses on

real industrial use cases that target sCPS. For the previous description, we focus on three basic aspects of sCPS, which are:

1. **Smartness.** This aspect represents the ability of the system to perform self-adaption, and to make decisions depending on the context. Basically, it is a match to using the feedback loop to control the behavior.
2. **Collective behavior.** This aspect allows entities to communicate with each other and coordinate behavior in a distributed fashion.
3. **Physical devices.** This aspect ensures that the system has a physical part, which means it has sensing and actuating capabilities. This aspect includes all such systems as Embedded system, Internet-of-Things and Cyber-Physical Systems.

1.2 Systematic Review

There are two types of systematic reviews that [18] mentioned, which are: mapping study and literature study (See Fig. 1[1]). Mapping study is a direct matching or analyzing of the extracted information, while literature study includes synthesizing a new knowledge or methodology based on extracted data.

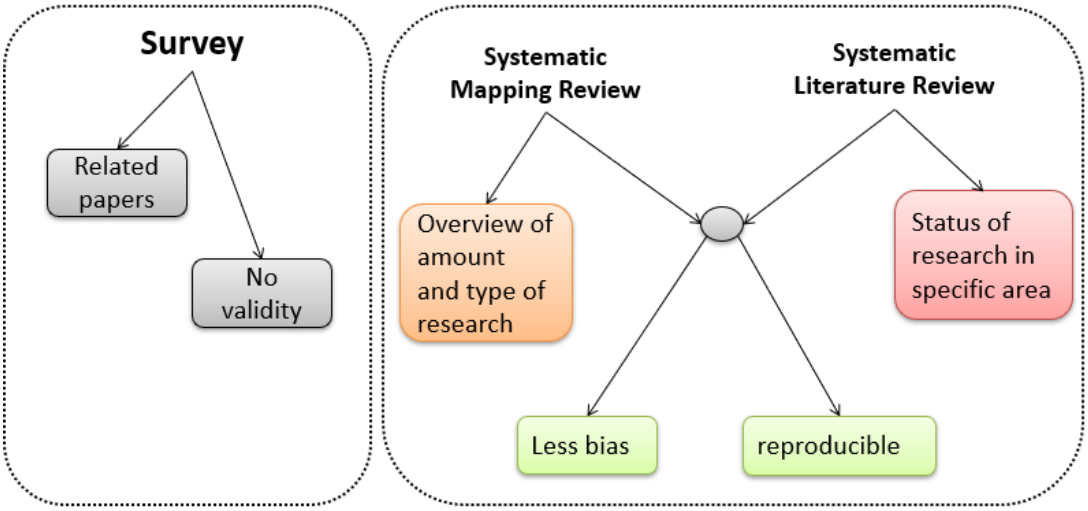


Figure 1: *The difference between Survey, Mapping Study and Systematic Literature Review*

The systematic review is split into three phases: planning phase, conducting phase and reporting phase. The planning phase includes definition of the research questions, and clarifies the source of primary studies. It also determines the inclusion and exclusion criteria, defines the study procedures, the quality assessment method, and the data extraction forms used to extract data relevant to the research questions. The planning phase results in a protocol that has to be approved by the researchers. The conducting phase details the application of individual steps described in the planning phase. Finally, the reporting phase provides a detailed description of the synthesized results.

The SLR guidelines suggest for the primary studies to be peer-reviewed papers. However, peer-reviewed papers does not guarantee the existence of real industrially-relevant demonstrators nor they provide a detailed description of relevant use cases that our study requires. This point was highlighted by the same author in another publication [17] where she mentioned that the type of research questions in the study determines what is the best selection as primary studies. We therefore chose deliverables of projects in place of peer-reviewed papers as primary studies.

By using the deliverables, we access both the results of their publications and an overall coherent description of the use cases. Also, we ensure to count the relevant use case just once even though it is mentioned in several publications.

The study (see Fig. 2) started with identifying the research questions, selection criteria and selection procedures, study quality assessment, data extraction forms, and data synthesis. The protocol was validated by internal researchers and external ones (i.e. published [2]). Then after conducting phase,

[1] <https://users.dcc.uchile.cl/~cgutierr/cursos/INV/CharlaSystematicReview.pdf>

we mention in the reporting phase a new realization about a guide to design uncertainty-aware self-adaptive component in CPS. This realization is build up on the knowledge extracted from the projects in the study.

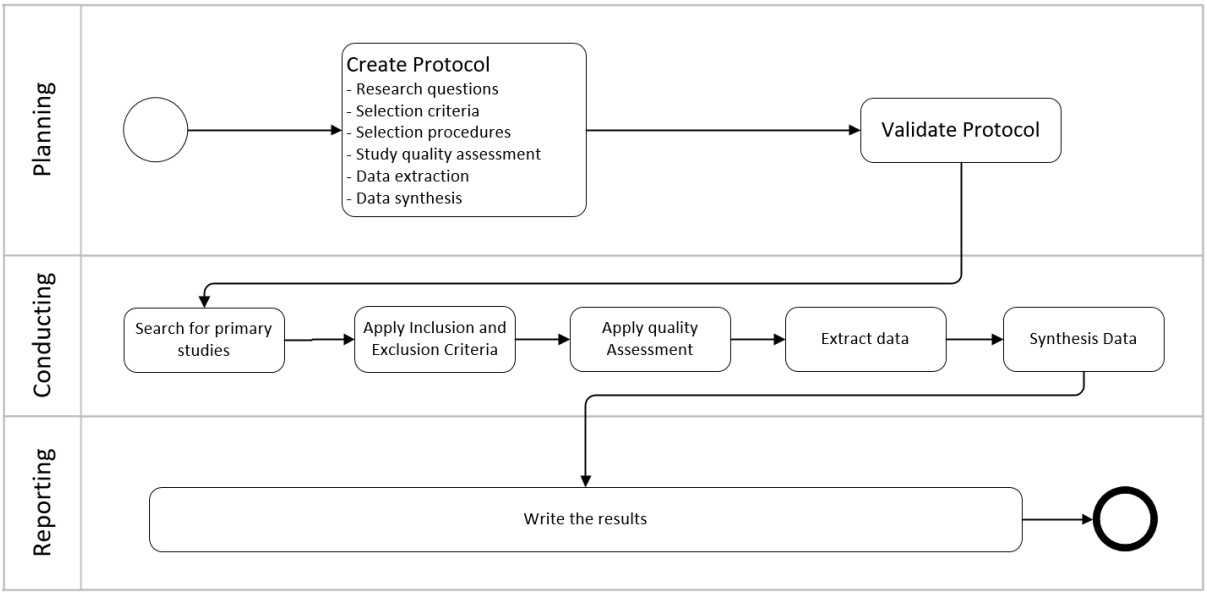


Figure 2: *The review steps*

The systematic review is split into three phases: planning phase, conducting phase, and reporting phase. The planning phase provides the background, defines the research questions, and clarifies the source of primary studies. It also determines the inclusion and exclusion criteria, defines the study procedures, the quality assessment method, and the data extraction forms used to extract data relevant to the research questions. The planning phase results in a protocol that has to be approved by the researchers. The conducting phase details the application of individual steps described in the planning phase. Finally, the reporting phase provides a detailed description of the synthesized results.

2 Planning

In this section we present the planning phase[2] of the systematic review, which are: research questions, search for primary studies, selection criteria, selection procedures, quality assessment, data extraction, and data synthesis.

2.1 Research Questions

Our study is driven by the following research questions:

RQ1 What are the typical domains for collective sCPS?

RQ2 What are the challenges addressed by collective sCPS?

RQ3 (updated)[3] What is the existing information related to uncertainty and self-adaptation in each demonstrator?

[2]The text is based on:
Rima Al-Ali: Industrial Use Cases of Cyber Physical Systems in EU Projects: Preliminary Study, in Position Papers of the FedCSIS 2017, Prague, Czech Republic, Polish Information Processing Society, pp. 187-193, DOI: 10.15439/2017F557, 2017.
Nevertheless, there are some small changes after the validation which required updating some points in the protocol. The points that had slight modification are associated with the word (updated).

[3]Added after publishing the position paper.

2.2 Search for primary studies

In the context of this study, the primary studies are represented by European projects with industrial leadership listed in the CORDIS database at the European Commission website[4]. Specifically, we target projects under the Information and Communication Technologies (ICT) calls. These are high-quality projects that report working (physical) demonstrator in their deliverables. When searching for a preliminary sample of related projects, the results showed that most projects returned by the query were accepted under ICT calls, confirming the validity of our selection for the calls. The CORDIS database can be considered trustworthy and provides results that are independent of commercial search engines.

We consider the projects from the past 10 years to provide a representative sample of recent results in this research area. Specifically, we targeted the FP7-ICT and Horizon2020-ICT calls [5]. The FP7 calls span years from 2007 to 2013, and the H2020 calls span years from 2014 to 2016. We did not include H2020 projects that started in 2016 nad later because these only have preliminary results.

2.3 Study selection criteria

We defined four basic selection criteria, shown in Tab. 1. Selection criteria 1, 2, and 3 contain a set of representative keywords, which we require to be included in the call and project descriptions. These keywords were extracted manually from a sample of related calls and project descriptions, and then filtered to obtain a more representative set.

Criteria Smart asp.	**Criteria 2 Collective asp.**	**Criteria 3 Physical asp.**	**Criteria 4 Entities**
smart(*)	distribut(*)	physical	$\# > 2$
intelligen(*)	de(-)central(*)	CPS	
adapt(*)	co(-)operat(*)	Internet of Things	
autonom(*)	communicat(*)	IoT	
aware(*)	collaborat(*)	embedded	
	connective(*)	device	
	emergent	hardware	
	swarm	robot	
	collective		

(*) means zero or several letters, (-) means it can contain a hyphen in the word, (#) means number of entities (i.e., physical nodes, human, ...) that interact in the use case.

Table 1
Selection Criteria

We include projects that satisfy *all* of the following inclusion criteria (IC):

Calls

IC1 The call was made in 2007–2016.

IC2 The title of a call contains at least one keyword from each selection criteria 1, 2, and 3 (c.f. Tab. 1).

IC3 The description of a call contains at least one keyword from each of the selection criteria 1, 2, and 3.

Projects

IC4 The project title contains at least one keyword from any of the selection criteria 1, 2, or 3.

IC5 The project description contains at least one keyword from each of the selection criteria 1, 2, and 3.

IC6 (updated) The project has deliverables.

Use cases

[4]CORDIS: `<https://cordis.europa.eu/>`

[5]European Commission. ICT work programmes 2007-2016, EU projects : `<https://cordis.europa.eu/guidancearchive_en.html>`

IC7 Challenges in the use case target collective behavior.

IC8 The use case has more than two interacting entities and at least 2 of them are nodes. The interaction represents the collective behavior (e.g., robots working together or interacting with humans). In this context, the entities could be nodes (e.g. robots) or humans.

We exclude projects that match *any* of the following exclusion criteria (EC):

Calls

EC1 Call started 2016-2017 since it is without final results yet.

EC2 Call description contains only keywords in context different from one of the three first criteria (e.g., collaboration between researchers).

Projects

EC3 Project description contains only keywords in context different from one of the three first criteria (e.g., collaboration between researchers).

EC4 (updated) Project does not have a working website – last time checked on 8/2/2018.

EC5 (updated) Project has no deliverables.

Use cases

EC6 Use case challenges do not target Application Layer nor Service Middleware Layer mentioned in [27]. For instance, uses cases that would be excluded target network reconfiguration at the link layer, or a collective knowledge exchange between people (e.g. social networks, software for rating places, roadmaps, and platforms).

EC7 Use cases have one-to-one or one-to-many relation between nodes such as robots and humans (e.g., wearables, tour guide robot).

2.4 Study selection procedures

The selection procedures are presented in four basic parts: primary studies, calls, projects, and use cases selections. The first part is related to the study foundation, and the procedure follows the guidelines in [18]. Therefore, we started with: (1) describing the background of the research interest, followed by (2) defining the research questions. Afterwards, (3) we selected the ICT calls 2007-2016 as a target (IC1). Next, (4) we defined four criteria out of which three are the sets of keywords (i.e. see Tab. 1). The second part is related to selection of ICT Calls, where (5) we selected calls with titles as mentioned in (IC2), (6) excluding the 2016-2017 calls (EC1). Afterwards, (7) we selected the calls with description in (IC3,EC2). Then, (8) we created a report containing all the selected projects. The third part is related to project selection, which starts with (9) including projects with titles in (IC4). Then, (10) we looked up for each project description in (IC5,EC3). Further, (11) we excluded projects that are without a working website (EC4). Then, (12) we selected the projects that had deliverables, or reports (IC6, EC5). Finally, (13) from the final set of projects, we created a list of all corresponding use cases. The fourth and the last part is related to selecting the use cases. Here, (14) we select the use cases with challenges that target collective behavior (IC7,EC6). Then, (15) we selected the use cases that are designed for more than two entities (IC8,EC7).

2.5 Study quality assessment checklists and procedures

We follow the selection procedures with evaluation of the quality of the selected demonstrators. Valid answers for the following check list items are: yes, partially, no. The list of related questions follows:

QA1 Is a challenge found in the deliverables missing from the project abstract?

QA2 (update) Does the challenge in the deliverables require a collective behavior and involve uncertainty in self-adaptation decision?

2.6 Data extraction strategy

We use the extraction forms as shown in Tab. 2, 3, and 4, Each table contains basic information that is needed to apply the selection procedures. The form for calls requires the title, the description of the call, and the fields for selection criteria 1, 2, and 3 which hold the data extracted from the project descriptions.

Data Items – Calls		
Call ID:	Call Title:	Call Description
Criteria 1:	Criteria 2:	Criteria 3:

Table 2
Data Extraction Form For Calls

Similarly, the second form for project data requires the title, description, website availability, and the fields for selection criteria 1, 2, and 3.

Data Items – Project		
Call ID:	Project ID:	Project Name:
Project Title:	Description:	Website:
Criteria 1:	Criteria 2:	Criteria 3:

Table 3
Data Extraction Form For Projects

The use case data form requires use case description, domain, challenges, number of nodes, and the availability of documentation. Finally, the targeted data to address our research questions is in the "Use Case Domain" and "Use Case Challenges". While the former is related to RQ1, the latter is related to RQ2.

Data Items – Use Cases		
Project ID:	Use Case ID:	Use Case Description
Demonstrator:	Domain:	Challenges:
Criteria 4:		
Demonstrator Description: Deliverables and Reports:		
QA1:	QA2:	

Table 4
Data Extraction Form For Use Cases

For extracting more detailed information related to uncertainty in self-adaptive systems, we answered the question of how the use cases tackle self-adaptation (Tab. 5) by answering the list of questions proposed in [21]. The questions are: When to adapt? Why do we have to adapt? Where do we have to implement a change? What kind of change is needed? Who has to perform the adaptation? How is the adaptation performed?

Data Items – Self-Adaptation					
When?*	Why?*	Where?*	What?*	Who?*	How?*

Table 5
Data Extraction Form For Self-Adaptation

Also, we list the questions related to uncertainty information (Tab. 6) that are relevant to the adaptation decisions. The questions are: What is the source of the uncertainty? What kind or type of uncertainty to tackle? What is the used method(s) to handle this uncertainty? What are the assumptions, inputs and outputs of the used method(s)?

Data Items – Uncertainty					
Source*	Type	Method(s)*	Assumptions*	Inputs*	Outputs*

Table 6
Data Extraction Form For Uncertainty

2.7 Data Synthesis

The data of the selected projects is extracted from the demonstrators. The synthesized data display the existing relations between the domains and challenges in each demonstrator. Furthermore, it illustrates the tackled uncertainties in them. Moreover, many different classifications are synthesized from the extracted data with demonstrating the information about the self-adaption. This information help in constructing a framework that provide the guidance for developers to design self-adaptive systems.

2.8 Approved Protocol

The protocol was validated by three members of the Department of Distributed and Dependable Systems. The participants have experience in different areas of software systems, which include 1) Smart Cyber-Physical Systems and IoT, 2) Dynamic Program Analysis, Measurement Methodology, Performance Modeling, and 3) Analysis of Dynamic Languages, Interpolation-based Verification Techniques. For further validation from external reviewers in the community, the protocol was submitted and published as a position paper [2]. In the second phase, the author suggested the extension and the details related to it, which was validated by the three members of the department as well.

To enhance the quality of results, we excluded some keywords from the first keyword selection, because for some keywords, the results proved too general. In addition, we added fourth criterion regarding the number of entities and requiring at least two of them as physical nodes. This is because a demonstrator needs to show the collective behavior where at least two physical nodes take a decision. We also made slight updates to inclusion and exclusion criteria specifying the exact starting time of a project, the available documentation, and the layers targeted by the challenges.

3 Conducting

This phase concerns applying the steps decided in the planning to perform the review and get the results. We illustrate in Tab. 7 the steps of the conducting with the resulted numbers of each selection procedure that starts with total number 266 of the presented calls in both FP7 and H2020.

After applying the selection procedures to calls using steps 1-8, the number of selected calls decreases to 30 calls, which corresponds to 384 projects. Then, after applying the steps 9-13, the number of the selected projects went down to 83 projects, which corresponds to 82 use cases. Finally, after applying the steps 14-15, the final number of selected use cases is 23.

Calls Title [266]	**Calls Description** [81]	**Project Title** [384]	**Project Description** [243]	**Use Cases** [82]
S[81]	S[30]	S[243]	S[83]	S[23]
				O[59]
			O[160]	
		O[141]		
	O[51]			
O[185]				

S stands for Selected, O stands for Others.
"S[*]" is the selected number of items depending on the name of column.
"O[*]" is the not selected items depending on the name of column.

Table 7
Selection of Use Cases

Later, we apply the quality assessment on the resulted set of use cases, we end up with the same number of projects and use cases. However, the number of challenges inside the use cases are different. The indicators are interpreted as following:

1. QA1 evaluation determines which challenges to count in the selection. So, if the evaluation is "yes" then the challenge is added. Otherwise, the challenge could be a part of another challenge partially or totally. Then, the evaluation is "partially" or "no", and the challenge is disregarded.

2. QA2 determines which challenge has uncertainty and requires collective behavior. So, if the evaluation is "no" or "partially" then disregard the challenge. The only evaluation to be considered is "yes", which has both uncertainty and collective behavior. Then, the challenge counts.

We extracted the data from the selected 12 projects and 23 use cases. To check the results of our protocol, we made an independent cross-check of the protocol on Horizon2020 calls of 2014-2015, and the extracted results were the same set of the selected use cases and challenges without the quality assessment.

We present synthesizing the dimensions that we will need for the mapping table between domains of the demonstrators in projects and their challenges.

1. Use Cases Domains (RQ1): The domains of the final set of use cases are counted after applying the selection procedures. It is worth mentioning that there is some an overlapping between those domains, but we relate each use case to just one domain. Exceptionally, VICINITY project mentioned more than one domain explicitly in its pilots, therefore we left it as they declared with counting a use case for each domain (i.e. instead of 4 use cases, it is counted as 8). The answer presents a classification of the domains presented in the projects.

2. Use Cases Challenges (RQ2): The challenges of the final set of use cases is counted after applying the selection procedures and the quality assessment. The challenges exist in the deliverables. It target uncertainty and collective behavior. We exclude from them the challenges that are not on application or service middleware layers. The answer for this question lies in a grouping of challenges under the corresponding properties, which illustrate the relation between the hazardous situations and properties.

3. Uncertainty Information (RQ3 (updated)): To answer this question, we matched the representative challenges of the selected use case to their domains first. Then, we stated the quality assessment on the mapping table, which shows which challenges are valid for each project in which domain. It also illustrates the uncertainties under challenges of each use case. More specifically, each challenge is decomposed to sub-challenges that are related directly to architecture. This helps in linking the targeted challenges in the use case to the uncertainties. The answer of this question includes classification of uncertainty types.

After going through the use cases, we conclude a set of points that are related to design involving uncertainty and updated the data extraction forms. The update included extending the considered aspects of the third question (RQ3*) as following:

1. Self-Adaptation Information: We list self-adaptation information in each use case.

2. Uncertainty Information: We list the uncertainty sources and types in each use case, their kinds and the methods to handle them. The answer of this question includes classification of methods that handle uncertainties and expected solutions for properties.

At the end, we propose a guide for developers that is based on the data from the use cases to help them in designing self-adaptive CPS taking into account the uncertainty.

4 Results

We will describe the situations that require self-adaptation in the projects and there relation to system properties, then we will explore the uncertainties there following the classification in [8] [24], and study the methods that are used to handle these uncertainties. Afterwards, we synthesize the relation between the system properties and uncertainty sources and required outputs so it is easy to match with the used methods.

After listing the selected projects, we start with studying the challenges and their hazardous situations. Hazardous situations create the need to adapt the system behavior or its structure to ensure the system dependability. These situations needs to be handled with considering the uncertainty involved

in the data provided for making the decision. Therefore, we study a set of industrial use cases[6] and extract the hazard situations, uncertainty sources and types and how they are handled there. The results provide us with enough examples for the developer that can be used in the methodology we present.

In this phase, we list the definitions of domains, challenges, projects, uncertainty source, uncertainty types, and methods used to handle uncertainty in industrial demonstrators in sCPS.

4.1 Projects

Tab. 8 lists basic information about the projects that we included in our study[7]. In case the project website is not accessible after the date in the protocol (projects denoted by "!" at its name), it is possible to access them using websites such as: https://timetravel.mementoweb.org/ or http://web.archive.org/. The list of projects follows:

Project Name	Project Title
T1: VICINITY (ID: 688467)	Open virtual neighborhood network to connect intelligent buildings and smart objects. Website: http://vicinity2020.eu/vicinity/
T2: symbIoTe (ID: 688156)	Symbiosis of smart objects across IoT environments. Website: https://www.symbiote-h2020.eu/
T3: CADDY (ID: 611373)	Cognitive autonomous diving buddy. Website: http://caddy-fp7.eu/
T4: OrPHEuS (ID: 608930)	OPtimising Hybrid Energy grids for smart citieS. Website: http://www.orpheus-project.eu/
T5: ClouT (ID: 608641)	ClouT: Cloud of Things for empowering the citizen clout in smart cities. Website: http://clout-project.eu/
T6: RECONFIG (ID: 600825)	Cognitive, Decentralized Coordination of Heterogeneous Multi-Robot Systems via Reconfigurable Task Planning. Website: http://www.reconfig.eu/
T7: HYDROBIONETS! (ID: 287613)	Autonomous Control of Large-scale Water Treatment Plants based on Self-Organized Wireless BioMEM Sensor and Actuator Networks. Website: http://www.hydrobionets.eu/
T8: INERTIA! (ID: 318216)	Integrating Active, Flexible and Responsive Tertiary Prosumers into a Smart Distribution Grid. Website: http://www.inertia-project.eu/inertia/
T9: NIFTi (ID: 247870)	Natural human-robot cooperation in dynamic environments. Website: http://www.nifti.eu/
T10: GreenCom! (ID: 318213)	MyGrid; Energy Efficient and Interoperable Smart Energy Systems for Local Communities. Website: http://www.greencom-project.eu/
T11: ASCENS (ID: 257414)	Autonomic Service-Component Ensembles. Website: http://www.ascens-ist.eu/
T12: IPAC! (ID: 224395)	Integrated Platform for Autonomic Computing. Website: http://ipac.di.uoa.gr/

Please access the projects with "!" using https://timetravel.mementoweb.org/ or http://web.archive.org/

Table 8
Projects of selected use cases

(T1) VICINITY The project presents a standard independent user-centric platform that provides interoperability between different IoT platforms in different domains. It allows users to share information in these domains, and the system to make better decisions in a decentralized way. This is achieved by using the concept of virtual neighborhood network.

(T2) symbIoTe The project aims at presenting a unified view on different IoT platforms. It can interoperate with each other in a secured way. The goal is to provide services for various smart environments,

[6] https://github.com/rima-alali/Uncertainty-Aware-Architecture/tree/master/SystematicReview
[7] More details about each project can be found in [3].

which contain sensors and actuators. Furthermore, these services are made available for end-users through mobile applications and web applications that are adaptive to the smart environment.

(T3) CADDY The project aims at helping divers in working, orientating and ensuring the diver's safety. To do so, the diver's physical state and behavior is monitored and the information forms input to decision-making process, in addition to the environmental conditions.

(T4) OrPHEuS The project presents collaborative hybrid-grid control strategies to minimize the waste in photovoltaic (PV) electricity production. The proposed solution considers new control strategies to transform overproduced electricity to energy for heating water or surfaces.

(T5) ClouT The project introduces a dynamic storage of historical data in a cloud that allows users to share services. Since the size of stored data from city IoT is unpredictable and huge, elastic and scalable clusters are required. On the other hand, sharing resources with many users at the same time may cause conflicts, which is also addressed by this project.

(T6) RECONFIG The project focuses on formal method-based planning for collaborative grasping and carrying objects by robots. The objectives of the projects are to improve dealing with visual information, robot-to-robot gesturing detection, point-to-point navigation, localization, stable grasp, and collaborative picking and carrying of items.

(T7) HYDROBIONETS The project aims at developing Wireless BioMEM Networks (WBNs), which have microbiological-awareness and are used for distributed monitoring of the water treatment and desalination process in a large-scale industry settings to improve the productivity.

(T8) INERTIA The project presents a system to manage efficiency of energy smart grids. It takes into account user comfort and provides dynamic pricing that minimizes expenses.

(T9) NIFTi The project targets cooperation between human and robots to rescue people. Urban Search and Rescue (USAR) involves Human-Robot Interaction (HRI) by spoken dialog and graphical user interface.

(T10) GreenCom The context of this project is Smart Grids and Energy. It aims at assuring the comfort of the users on one hand, and saving energy consumption on the other one. This is done by monitoring and supporting decision-making of customers. Hence, the targeted challenge lies in having an efficient energy management that enhances performance while preserving security.

(T11) ASCENS The project targets designing dynamic systems with collective behavior. The aim of those systems is to form a group able to solve complex problems that an individual is not able to. Therefore, a methodology that targets emergent behavior engineering is presented. It focuses on self-adaptation and awareness issues in addition to performance.

(T12) IPAC The project presents a platform for building collaborative autonomic embedded systems on mobile nodes. The proposed middleware is used for developing services that are context-aware and use short-range communication for information exchange (i.e., event-based). Thus, the used information dissemination algorithms are similar to how rumor spreading works.

4.2 Domains Classification

We define a set of domains that classify the use cases (see Tab. 9) including explanation of each domain. We illustrate the numbers of the use cases and their selection (Fig. 3) and the percentage of the selected use case for each domain (Fig. 4).

More specifically, we partially considered the classification used in [13] with small modifications and extensions to capture the mentioned domains in our study. The list of particular domains follows:

Domain	Example
D1: Smart Factory and Manufacturing	automation of monitoring and tracking products, manufacturing boards
D2: Smart Traffic and Transport	smart parking, navigation
D3: Smart Homes and Buildings	efficient energy and gas use
D4: Smart Grids and Energy	decrease energy waste, renewable resources
D5: Robotics	cognitive robots, monitoring robots
D6: Crisis and Emergency	rescue unmanned robots
D7: Cloud	service and resources provider
D8: Telecommunication	local customized mobile network
D9: Healthcare and Medicine	ambient assistant robots, surgery robot, wearable health monitoring devices

Table 9
Domain examples

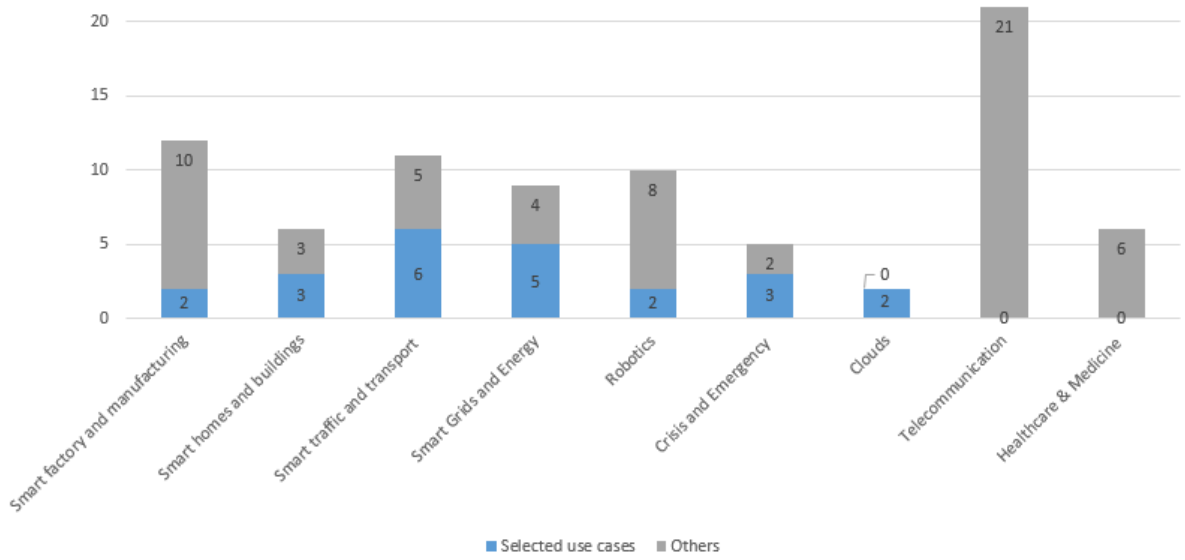


Figure 3: *The numbers of use cases in each domains*

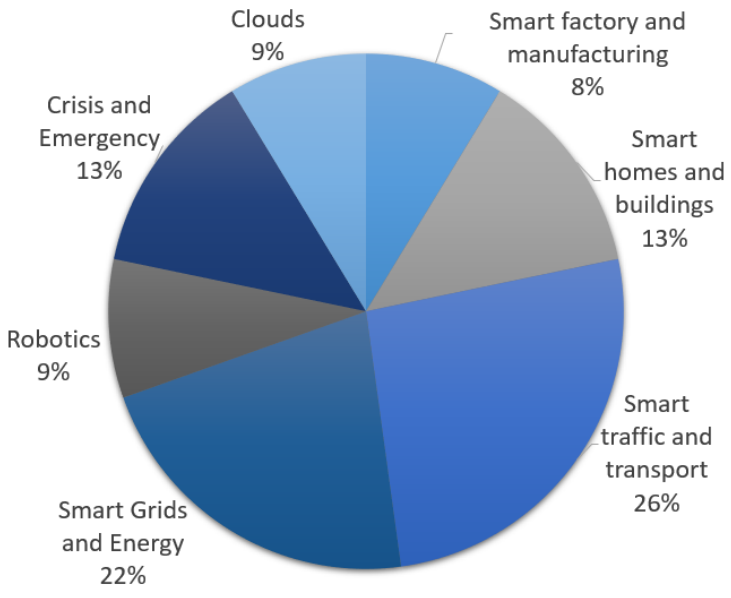


Figure 4: *The percentage of selected use cases for each domains*

(D1) Smart Factory and Manufacturing This domain includes all the smart industry aspects from smart technologies in production to manufacturing products. Furthermore, smart industries and warehouses use monitoring, tracking products, and quality control of production process (e.g. monitoring water/gas/oil distribution network). Therefore, the industry automation increases efficiency and reduces costs. In addition, this includes manufacturing new products and their customization such as new boards and energy batteries.

(D2) Smart Traffic and Transport This domain consists of two basic parts, which are (semi-) autonomous vehicles/plains/trains and smart management of traffic. The automation in vehicles allows them to navigate, self-drive and smart-park, while, as to the traffic management, it provides an infrastructure for predicting traffic and support for smart highways.

(D3) Smart Homes and Buildings This domain includes the automation of urban spaces such as homes, offices and buildings. The automation requires having communication infrastructure known as Internet of Things (IoT). It consists of a network of sensors and actuators for monitoring and decision-making purposes. The aim is ensuring optimized and efficient use of resources (e.g., energy) in addition to safety.

(D4) Smart Grids and Energy This domain describes management of smart grids with the aim to keep the balance between the demand and the generated energy. Furthermore, the renewable energy is the future technology where the electricity is generated from environment-friendly sources and the system rises the customer awareness about his energy consumption and provides him with the ability to customize the services.

(D5) Robotics This domain contains developing robots that are cognitive or mimic the behavior of living organisms. As well as unmanned robots that co-operate with humans. For instance, a robot monitors the environment to help human in navigation underwater.

(D6) Crisis and Emergency This domain contains the devices and robots that help in case of crisis by forming teams with humans in rescuing tasks.

(D7) Clouds This domain presents clouds as a provider for services and resources (i.e. software and hardware) for IoT entities and users.

(D8) Telecommunication This domain consists of connected mobile or communication devices with a possible internet connectivity. These connected devices provide customized services and allow for network reconfiguration. In case of mobiles, for instance, a provided service for the users could be a local tour guide application, or just knowledge exchange between the users as in social networks.

(D9) Healthcare and Medicine This domain contains the ambient assistant systems, medical devices, and health management networks. For instance, the ambient assistant robots that help elderly people to walk, surgery robot, and wearable devices to monitor peoples health such as blood pressure.

4.3 Challenges Classification

We start with studying hazardous situations. Hazardous situations give rise to the need for adapting the system behaviour or its structure. These situations need to be handled with considering the uncertainty involved in the data provided. Thus, we extract the hazard situations, uncertainty sources and types, and how they are handled there.

The presented challenges in the use cases are strongly associated to a required property in the system. Therefore, we grouped the mentioned challenges under the related property in Tab. 10.

We define the properties associated to the challenge groups that are tackled by the demonstrators in this study as follows:

Property	Hazard	Challenge	Example
P1: Safety	Human body functions failure	Dangers from unexpected environment	Detect human physical state [T3]
P2: Performance	Lack of resources or inefficient resources management	Optimize resources usage and learn context and human behavior	Hybrid energy grid [T4], Sharing resources in the cloud [T11]
P3: Price	Inefficient resources management	Smart management for resources	Add renewable energy generators [T1], Minimize cost by learning from tariff and resources consumption [T10].
P4: Resilience	Component Failure	Overcome failures	Sensors classification faults statistically [T5], Overcome failure of an individual by forming new ensemble [T11], Spotting new chair in the room that was not there [T6]
P5: Cooperative	Manipulated situations and inefficiency in achieving the goal by individual	Unexpected and changing context and goal achievement requirements	Robot-human collaboration in diving [T3], Rescue mission by a team of robots [T11]
P6: Connectivity	Communication limitations	Limited Communication with collisions	Short range of communication between sensors in the industry [T7]
P7: Authorization	Unauthorized access or manual process	Authorization for accessing physical facilities or data	Authority of multi-sensory network in homes [T10]

Table 10
Properties and Challenges

(P1) Safety This property includes the challenges that target avoiding dangerous situations for human or unwanted circumstances in a collective manner, such as dangerous health condition [T3] and dangers from being in unexpected environments such as detecting intruder [T2]. Another example is avoiding vehicle accidents [T12].

(P2) Performance This property targets the challenges related to managing resources in the system. More specifically, the resources include many aspects such as hardware, time, energy, ... etc. For instance, energy management [T1][T2][T4][T8][T10][T11] is an important issue especially in Smart Grids since providing a better demand and response strategies can hugely help in decreasing energy waste [T4]. Additionally, load balancing and sharing resources in case of cloud can improve the performance and the scalability [T11]. Also, issues related to utilizing the buildings and parking slots usage [T1] are considered, in addition to optimizing routes. Tracking the quality of a service or a product is tackled in [T7] that considers the water treatment during desalination process and [T12] during production process.

(P3) Price This property targets the challenges that consider the financial aspect of the demonstrator. Surely, applying any new technology in real-life has the cost of the hardware, the installation and the running as one of the main concerns; therefore, it should be reasonably balanced with the other properties. For instance, using renewable energy generation minimizes the cost of energy generation [T1], smart energy management [T10], and transforming the energy to different forms to reduce the waste [T4]. Also, utilizing the usage of resources reduces the costs such as stopping Virtual Machines (VM) after being idle for a while [T11].

(P4) Resilience This property includes the challenges that target dealing with failures of some components [T2][T3][T11], abnormal behavior [T7], resolving conflicts [T5], and handling unexpected or manipulative situations by increasing system awareness to its context [T6]. Thus, it requires that the system features a level of dynamicity that allows it to adapt to the current circumstances that might include humans as a part of the system [T10]. Most importantly, this dynamicity mainly requires a collective behavior to overcome or adapt to the context changes and recover from the failures in such a way that the system will still be able to provide the services to the end users.

(P5) Cooperation This property includes the challenges that target the group behavior of system components, which aims at providing required services. For instance, robots exchange information

to achieve shared goals such as moving objects [T6], rescue missions [T9] (with human involvement [T11]), avoiding collisions [T9], finding parking lots [T12], and robot-human collaboration in diving [T3].

(P6) Connectivity This property includes the challenges that target communication problems including limitations in communications [T6], short ranges [T7], and network problem [T12]. Hence, the system tries to handle those issues collectively.

(P7) Authorization This property includes the challenges that target authority issue in the systems to avoid criminal actions such as hacking inside a home sensor network that could cause privacy violations and hacking related to vital resources management system [T10]. Additionally, handling physical access issues to facilities such as ports are also considered [T2].

5 Answering Self-Adaptation Questions

The classification is based on self-adaptation questions in [21] as following (see Tab. 11):

Project	Domains	Properties	When?	Why?	Where?	What?	Who?	How?
VICINITY	D2, D3, D4	P2, P3	Proactive, Reactive	User, resources	E, M	Parameter, Structure	Node, Cloud	Decentralized, Centralized, Rules
symbIoT	D2, D3	P1, P2, P4	Proactive, Reactive	Context, resources	E, M	Parameter, Structure	Node	Decentralized, Rules
CADDY	D5	P1, P4, P5	Reactive	User, resources	E	Parameter	Node	Decentralized
OrPHEuS	D4	P2, P3	Proactive	Resources	E	Parameter, Structure	Cloud, Edge	Decentralized semi-centralized and centralized, rules
ClouT	D7	P4	Reactive	Resources	E, M	Parameter	CPaaS Layer	Decentralized, rules
RECONFIG	D5	P4, P5, P6	Reactive	Resources	E	Parameter	Node	Decentralized
HYDROBIONETS	D1	P3, P4, P6	Proactive, Reactive	Context, resources	E	Parameter, Context	Gateways, Micro-servers, Nodes	Centralized, Cluster, Distributed
INERTIA	D4	P2	Proactive	Resources, user	E, M	Parameters, Context	Hubs	Decentralized, Rule-based decisions
NiFTi	D6	P4, P5	Proactive, Reactive	Context, user	E	Parameter	Node	Decentralized
GreenCom	D4	P2, P3, P4	Proactive	Context	E	Structure	Microgrid	Decentralized, rules and policies
ASCENS	D3, D6, D7	P2, P3, P4, P5	Proactive, Reactive	Context, resources	E	Parameter, Structure	Node	Decentralized
IPAC	D1, D3, D6	P1, P2, P5, P6	Reactive	Context, resources	E, M	Structure	Node	Decentralized, rules and policies

Table 11

Self-Adaptation Information

Domains - D1: Smart Factory and Manufacturing, D2: Smart Homes and Buildings, D3: Smart Traffic and Transport, D4: Smart Grids and Energy, D5: Robotics, D6: Crisis and Emergency, D7: Clouds;
Properties - P1: Safety, P2: Performance, P3: Price, P4: Resilience, P5: Cooperative, P6: Connectivity;
Why? resources: Change in the technical resources; context: Change in the context; user: Changes caused by user(s);
Where? E: Ensemble of Applications, M: Middleware

The study shows a set of projects that have demonstrators in various domains, where most of the systems are proactive when it comes to the time of performing the adaptation. Nevertheless, there are a number of systems that are reactive and depend on estimation or optimization of data rather than predicting the situation. The reason of adaptation is mostly from changes in technical resources. The studied systems have Ensemble of Applications where the adaptation is mostly performed using parameter or structure techniques at the node, middleware, edge, or cloud level. Notable, Service and Middleware layers were already the target in the study as well as having decentralized systems. Nevertheless, there was a case of centralized adaptation such as in cloud (i.e. OrPHEuS project has the adaptation in two levels centralized and decentralized).

For the parameter technique, the adaptation decision keeps the same structure, while structure technique is more for reconfiguration or forming dynamic groups (i.e. ensembles). The context technique depends on the type of effectors used in the demonstrators that are affected by the adaptation.

5.1 Uncertainty Sources Classification

Depending on [8] and [24], a classification of uncertainty sources that are in the industrial use cases [2] are presented in Tab. 12 (i.e. the existing uncertainty sources in the selected project are in bold).

More specifically, the uncertainty sources targeted in the projects are:

Variability space of adaptation This source is about size of variability space involved in the adaptation decision such as monitoring human physical state (i.e., vital parameters) [T3].

Classification	Source
Model uncertainty	Abstraction Incompleteness Model drift Different sources of information Complex models
Adaptation functions uncertainty	**Variability space of adaptation** [T3] Sensing **Effecting** [T7] **Automatic Learning** [T1] [T2] [T3] [T4] [T5] [T6] [T7] [T8] [T9] [T10] [T11] **Decentralization** [T4] [T11] [T12] **Changes in adaptation mechanisms** [T2] **Fault localization and identification** [T3] [T6] [T9] [T12]
Goals uncertainty	Goal dependencies Future goal changes Future new goals Goal specification Outdated goals
Environment uncertainty	**Execution context** [T9] **Human in the loop** [T9] **Multiple ownership** [T3]
Resources uncertainty	New resources **Changing resources** [T2] [T4] [T6] [T7] [T9] [T11] [T12]
Managed system uncertainty	System complexity and changes

Table 12
The Classification of Uncertainty Sources in Studied Industrial Use Cases

Effecting This source of uncertainty is related to unpredictable behavior of effectors such as handling abnormal behavior and malfunctions [T7].

Automatic learning Here, the source of uncertainty is related to used technique in making decisions such as machine learning and statistical approaches, which include some amount of errors during their calculations to find the fitting models. For instance, machine learning or statistical approaches is used for efficient resources and cost management [T1][T2][T4][T8][T10][T11], learning behavior as in case of bacteria growth [T7] or room occupancy [T2], recognition as in case of robot recognizes the diver's gestures [T3] or finding all chairs in a room full of chairs and tables and moving them to another room [T6], classification as in case of faulty sensors in the aim of balancing load [T5], and avoid robots collisions by increasing situation awareness [T9].

Decentralized This source of uncertainty here is related to entities taking decision collectively such as exchange information between vehicles to avoid accidents [T12], using hybrid energy grid to distribute the energy in different forms [T4] and robots cooperate to perform a rescue mission [T11].

Changes in adaptation mechanisms The source of uncertainty here is related to changing the infrastructure according to changing the goals such as triggering and driving temperature from another room in case of coil failure [T2].

Fault localization and identification The source is related to noisy measurements or identifying faults. We added the noisy measurement to this part since in the projects the parts are mostly related to localization or identifying objects. For instance, it is possible to do classification of sensor faults to balance load [T5], avoid robots collisions by increasing situation awareness [T9], robot estimates his position [T3], finding all chairs in a room [T6] and finding parking for the vehicle [T12].

Changing resources The source here is related to dynamicity of resources such as avoid resources lose as in case of using hybrid energy grid to distribute the energy in different forms [T4], optimizing VM usage [T11] and energy consumption [T11][T12], failures [T2][T11], limitations in communication

[T6][T7][T12], abnormal behavior and malfunctions [T7], or unexpected environment as in case spotting new chair in the room that was not there [T6] and Robot-Human cooperation to perform a rescue mission [T9],

Human in the loop This source is related to human behavior when he/she is involved in the decision making process such as Robot-Human cooperation to perform a rescue mission [T9]

Execution context This sources is related to the unexpected environment such as avoiding robots collisions by increasing situation awareness [T9].

Multiple ownership This source is related to conflicting goals such as conflicting tasks for a robot [T3].

5.2 Uncertainty Types Classification

In addition to the uncertainty sources, we need to recognize the uncertainty types that the use cases include. We here list the uncertainty types related to each source in Tab. 13, where we divided uncertainty types into categories as follows:

Uncertainty Classification	Uncertainty Source	Uncertainty Type
[SC1] Adaptation functions uncertainty	[S1] Variability space of adaptation	U5
	[S3] Effecting	U6
	[S4] Automatic Learning	U2, U3, U4, U5, U6, U7, U8
	[S5] Decentralization	U1, U6, U7, U8
	[S6] Changes in adaptation mechanisms	U6
	[S7] Fault localization and identification	U2, U3, U8
[SC2] Environment uncertainty	[S10] Execution context	U8
	[S11] Human in the loop	U8
[SC3] Resources uncertainty	[S12] Changing resources	U1, U2, U3, U4, U6, U8

Table 13
Existing uncertainty types under each uncertainty source classification

(U1) Network and Delays This category is related to end-to-end network communication problems [T4][T6][T9][T12], such as delays [T6][T12] and offloading latency [T4], in addition to cases of having limited [T9] or no explicit communication [T9]. (i.e. we consider the noise in the communication as a separate type)

(U2) Missing information This category is related to lack of knowledge [T4], losing information [T9][T12], lack of detection [T2] or missing information due to malfunction of a part of the system [T3][T7].

(U3) Noise This category is related to noise in measurements [T2], communication [T7] or processing noisy images [T3], inaccurate localization [T3][T6][T9][T12] and pointing or observation [T3][T6].

(U4) Peeks This category is related to keeping the values under a certain threshold. For instance, efficient resource management such as energy [T1][T2][T8], road planning [T2], parking space [T1] or hardware resources [T11] requires dealing with peaks in the usage, also keeping the efficient cost management of used resources [T1][T4][T10][T11].

(U5) Ambiguity and Ill-definition This category is related to problems in defining the models such as bacteria functioning [T7], recognizing bad gestures [T3] or objects [T6], risk from dynamic features as in heart attacks [T3], or sensors fault classification [T5].

(U6) Failures This category is related to failing of a component/part of the system or abnormal behavior [T2][T3][T5][T6][T7][T11].

(U7) Inconsistency This category is related to verifying the local evaluation [T12] and avoiding collisions [T3][T7][T9] by resolving inconsistency or conflicts.

(U8) Other uncertainties related to context This category is related to photo-voltaic (PV) energy surplus/overvoltage [T4], unknown environment [T9], flexibility in energy management by context awareness [T4][T10], such as in case of detecting the comfort level for occupants in a room [T1][T8], and mental occupancy of a rescuer [T9], and avoiding obstacles [T11].

As the Tab. 13 shows, the uncertainty types are not specific to a source of uncertainty, the same type might exist in different sources.

5.3 Mapping Domains and Properties

Tab. 14 illustrates the relation between each property and use case of the projects. It worth mentioning that in the case there are more than one use case in the same domain in the same project, we merge them together in the table.

Domain	Project Name	Safety	Performance	Price	Resilience	Cooperation	Connectivity	Authorization
Smart Factory and Manufact.	HYDROBIONETS		U5		U2,U3,U6		U1,U3,U6	
	IPAC		U1,U2			U3		
Smart Homes and Buildings	VICINITY		U4	U4				
	symbIoTe		U4		U2, U3, U6			
	VICINITY		U4	U4				
Smart Traffic and Transport	symbIoTe		U4		U2, U6			
	ASCENS		U4					
	IPAC	U1,U7						
Smart Grids and Energy	OrPHEuS		U1,U2,U4,U8	U4				
	GreenCom		U8	U4	U8			
	INERTIA		U4,U8					
	VICINITY		U4,U8	U4, U8				
Robotics	CADDY	U5			U3,U5,U7	U3,U5,U7		
	RECONFIG				U3, U6	U3,U5	U1	
Crisis and Emergency	NiFTi				U3,U7,U8	U1,U2,U3,U8		
	ASCENS				U6,U8	U6		
	IPAC						U1	
Clouds	ClouT				U5, U6			
	ASCENS		U4	U4	U4,U6			

Table 14
Selection of Use Cases
Light gray: before QA1, dark gray: after QA2, no uncertainty : empty gray cell

The table contains the targeted challenges with the existing uncertainties mentioned the demonstrators. The results show that in Robotics and Crisis & Emergency domains, the research does not focus on performance, cost or authorization issues, while the rest of domains take performance and cost into account as two of the main challenges. The performance and cost are usually coupled together, which makes the challenge mostly related to realizing the context and managing resources. In case of Robotics and Crisis & Emergency, the focus mostly lies on resilience and cooperation to achieve a common goal, where failure is the main issue there. Connectivity has network problems and delays as the main challenge and safety challenges vary between delays, inconsistency and ambiguity. Later on, we will skip mentioning the authorization, since there was no collective behavior in addressing the challenge.

We notice many points that are candidates to form a relation between uncertainties and architecture in addition to self-adaptation (i.e. for us: self-adaptation is triggered by hazardous situations that are associated to system properties as mentioned previously in Sect. 4.3, which are:

System Features The features of a system determine the uncertainties that need to be dealt with. Understanding these relations helps deducing how to handle the uncertainties. More specifically:

- Selecting the Domain. The domain may impact the selection of the level where the computation is done, the property of interest and the types of uncertainties and models. For instance, in the Smart Factory & Manufacturing domain, uncertainty tends to be handled with the least effort possible, e.g., deactivating a malfunctioned sensor, or monitoring entities that have neither models nor assumptions.
- Selecting the level where to perform the computation. This selection impacts designing the system such as handling noise and ambiguity could be done on the local node. Choosing a Distributed/Peer-to-Peer (D/P2P) architecture implies that delays should be handled in addition to other possible uncertainties.
- Selecting the Property. The property can be coupled with another property in a trade-off relationship. This influences selecting the uncertainties and their models. For instance, having performance in mind, the noise is handled in resilience using subspace projection or in connectivity using more measurements.

Models and Assumptions The same models that are used in general cases could be enhanced by adding different parameters. For instance, it is better to use trajectory instead of a position and speed when following a robot, i.e., not relying on communication but visual recognition, or choosing a special combination of sensors to enhance the accuracy of measurements. Another case is to overcome model limitations by combining it with another model. For instance, it is possible to predict occupancy depending on historical data by creating a Transition Matrix based on Markov Chains (MC) (i.e. MC remembers the previous state only).

Design/Runtime Most of the uncertainties are handled by models at runtime. Nevertheless, there are design time solutions related to noise, ambiguity, or delays. Simply, the solutions are in deciding upon the kind and number of sensors involved in monitoring or the type of communication (e.g. broadcasting). These solutions are extended in other use cases to runtime reconfiguration solution (e.g. communication interfaces reconfiguration).

Human Intervention and Degree of Autonomy Some domains prefer to keep human involvement such as in Crisis & Emergency for psychological reasons (i.e. avoid designing fully autonomic robots because of trust issues).

Relations between Uncertainties For the developer, it is not enough to be aware of the uncertainty types that might influence the chosen system property. They need also to consider the relations between them. For instance, safety property is influenced by delays in D/P2P architectures. This uncertainty is related to inconsistency in data, which also influences safety. On the other hand, it is related to missing information and both influence performance. In case of delays, the approach to handle it is to reconfigure the communication interface, which also helps in handling inconsistency. For performance, the solution is based on using broadcasts to minimize the delay and handle missing data by lowering the channel sampling.

Another example is related to the performance property. The first case is to handle delays and missing information alone in D/P2P level, and the other case is delays, missing information, threshold and context in edge level. In the first case the delay is handled by architectural decisions using broadcasting, while in the second case by using Stream Processing Framework. While the missing information in the first case is minimized using lower channel sampling, in the second case interpolation and prediction is used because threshold and context are also involved in the use case. In other words, dealing with peaks needs a model of values for the evaluation. Choosing the other solution (i.e. lower channel sampling) means less values, which impacts the model values and the final evaluation.

We can conclude that these different aspects have to be considered during the design. For instance, taking an example from outside the projects, the authors in [12] emphasize the challenges of introducing robots and AI to telecommunication similarly to other industries, and propose an interesting analysis for the current application and their benefits. For instance, the report presented the usage of robots in service automation such as customer service and back-office tasks. Even though the projects in our study do not cover telecommunication domain, we still can match nicely the proposed applications there to our study. More specifically, the applications target resilience (i.e. avoiding failures by proactive recognition), cost (i.e. predictive network maintenance), and performance (i.e. service management

efficiency). Based on the Tab. 14, it will be interesting to investigate other properties that Robotics domain could bring to Telecommunication. For instance, safety of human as a customer service might be provided by establishing a virtual neighbourhood network (e.g. connect hospitals, drones, doctors and patients in the same area to deal with emergency health cases). Additionally, considering resilience, cooperation and connectivity properties could result with new possible applications. For instance, the drones or the robots can form an edge in the network to recover from a failure or reduce latency. It is worth mentioning that our results are not exclusive to learning, so using AI is one of the possible approaches as we will discuss later.

5.4 Methods for Handling Uncertainty Classification

The main points that the developer should consider are in choosing the method to handle uncertainties illustrated in Fig. 5 and can be captured by the following questions:

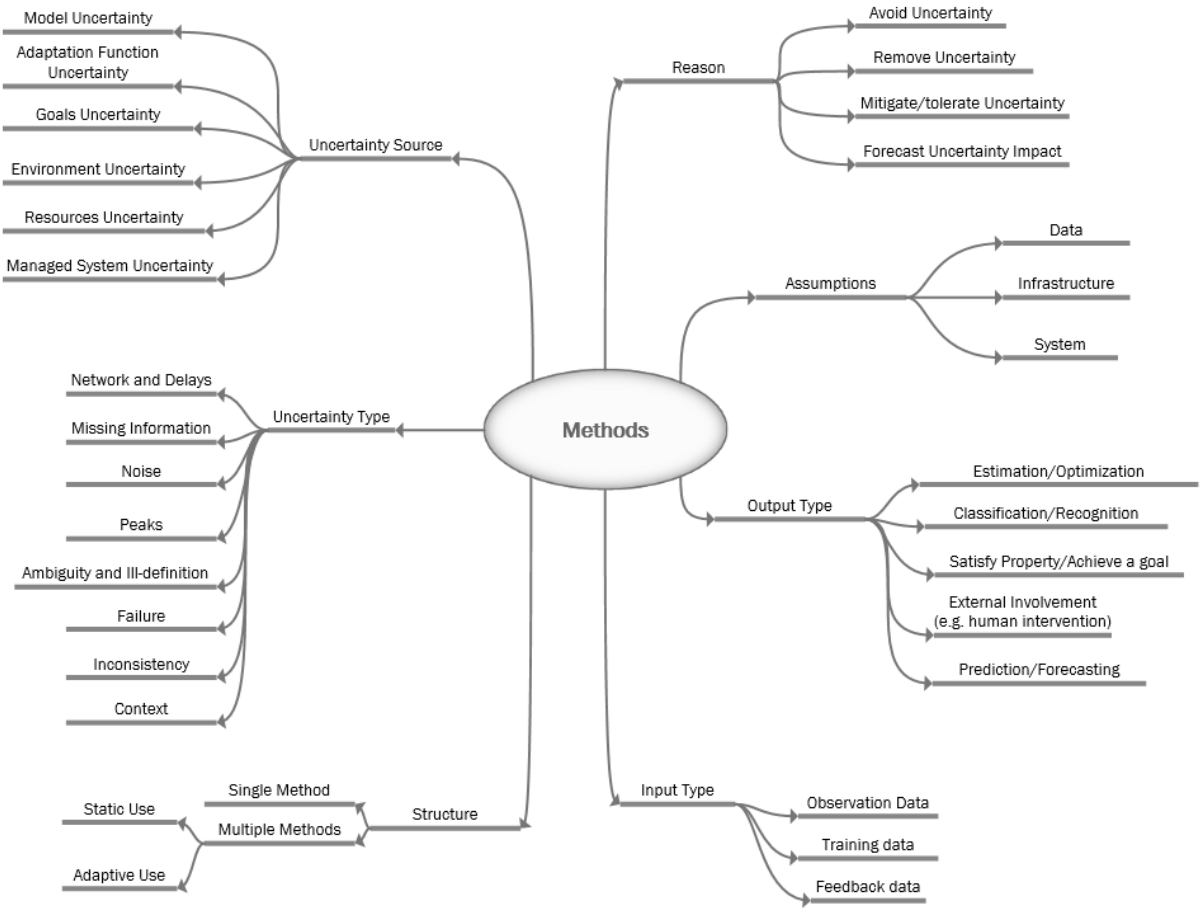


Figure 5: *Taxonomy of Methods to Handle Uncertainty*

1. What are the sources of uncertainties handled by the method?
2. What are the uncertainty types that the method handles?
3. What is the reason to use the method?
4. What are the output types of the method?
5. What is the structure of the method(s) used?
6. What are the assumptions of the method?
7. What are the input types of the method?

Uncertainty Source As this part is explained previously, it includes the various causes of uncertainties, which helps the developer to consider the different possible issues. The uncertainty source might be on modeling, adaptation function, goals, environment, resource or managed system levels.

Type of Uncertainty Methods are developed to handle (a) specific type(s) of uncertainty. In case of having many options for the same targeted uncertainty type(s), the selection of the method mostly depends on the data, their use, the impact of this kind of uncertainty on the overall system operation and efficiency reasons such as resources limitation and financial aspects. For instance, having high accuracy of outside temperature measurements used for weather forecasting is less important than measuring the distance between two vehicles to avoid an accident. The latter one requires more accuracy and frequency of readings, which means higher quality of hardware and consequently higher price to get better results in localization.

Reason Here, the reasons to use a specific method as an approach to deal with uncertainties is investigated. Learning from engineering dependable systems, four approaches to deal with faults are mentioned in [19]. Similarly, we list the reasons behind using the method, which are: avoiding uncertainty, removing uncertainty, mitigating/tolerating uncertainty, and forecasting the uncertainty impact. Both avoiding and removing uncertainty if possible is done during the system development, while the mitigating/tolerating the uncertainty could be applied afterwards during system operation. Forecasting of uncertainty impact might be done in cases in which the other approaches are not applicable. For instance, various types of sensors can be added to increase the accuracy of measurements, which helps in avoiding fault of localization. An example of removing uncertainty is dropping a faulty sensor when it has an abnormal behavior. The mitigation of uncertainty is present in recognition of gestures in noisy images. Finally, forecasting the uncertainty impact is captured in cases of predicting the peaks in energy consumption.

Structure During choosing the methods to handle uncertainties, the developer should take into account how the methods are used if there is more than one. For several methods, there are static use or adaptive use (similarly to static and dynamic redundancy idea proposed in [19]). For instance, a single method mitigate uncertainty by modifying related parameters such as lower the sampling time in case of minimizing missing information, or by changing the structure such as changing communication interfaces in case of big delays. As for static use of methods, it is possible to predict occupancy depending on historical data by creating a Transition Matrix based on Markov Chains (MC) (i.e. MC remembers the previous state only). Adaptive use of methods as mentioned in OrPHEuS in this publication [14] proposed to have reinforcement learning that learns from outcomes of supervised learning methods for the home energy usage. This type of methods usually responsible for updating the threshold that controls adaptation decision making process, or having weights associated with the feedback loop.

Assumptions Each method has its own assumptions. Depending on the available data, needed output and the scenario, the assumptions can be limiting. For instance, using Markov chains is not a suitable method when a trend over historical data is the target. The assumptions can be associated to data, infrastructure and the system itself. For data, the assumptions are about the involved variables such as being (in-)dependent variables, their distribution or associated weight (e.g. this is used when voting over many sources for values). Another example is the time series components (i.e. trend, seasonality, correlation and multi-variant). Regarding the infrastructure, the assumptions are associated to resources such as the hardware configuration or limitation of communication (e.g. bandwidth or short range communication). As for the system itself, examples of its assumptions are the system model and the probability associated to its states/transitions. Also, the fact whether the system is (non-)linear or (non-)deterministic, which impacts hugely the choice of suitable methods.

Output Type The output type determines the targeted results from using the method. Therefore, it is important to consider this part before going for further details. Here, we list the expected results from the methods covered by the study, which includes: estimation/optimization (i.e. includes learning when input is feedback data), classification/recognition, satisfy property/achieve goals, prediction/ forecasting and external involvement.

Input Type The inputs to the methods can be observation data such as historical data, training data such as a (un-)labeled training data, and feedback data such as in control loops or involvement of human.

We presented the used methods in the projects to manage the uncertainty types inspired by [8]. The classification is presented in Tab. 15.

Method Classification	Technique	Method Example
HU1: Data Filtering and Estimation	Projection	Subspace Projection [T7]
	Estimation	Kalman Filter [T6]
	Filling Data	Matrix Completion [T7]
HU2: Statistics and Probability	Frequencies	(S)ARMA [T5]
	Probability	Fuzzy Logic [T3]
	Markov process	Hidden Markov Model, Semi-Markov [T7]
HU3: (Re-)Configuration	Design Time	Static adding for more sensors during the design time [T3]
	Runtime	Dynamically change of communication interfaces [T12]
HU4: Machine Learning & Neural Networks	Supervised Learning	Neural network [T7]
	Unsupervised Learning	k-means*
	Reinforcement Learning	Q-learning*
HU5: Verification	Design Time	Hybrid Automata*
	Runtime	SMC-BIP[T11]
HU6: Human-in-the-loop	Total Involvement	Human takes over: User Notification/Alarm [T10]
	Partial Involvement	Cooperation with human [T9]
HU7: Declarative Programming	Constraint programming	Ensemble (i.e. CSPs) [T11]
	Modeling	Behavioral Trees [T6]
	Logic programming	Prolog [T12]

Table 15
Methods Classification
*The text in gray is there for illustration and not mentioned explicitly in the deliverables.

(HU1) Data Filtering and Estimation This group contains three basic categories, which are: projection of the variable space to a smaller dimension (U3 - T7), estimation such as in case of data fusion (U3,U7,U8 - T9) (U2, U3, U7 - T12) interpolation (U4 - T2)(U2, U6 - T2) or filtration (U1 - T6)(U1 - T9), and filling data such as matrix completion (U2 - T7).

(HU2) Statistics and Probability This group contains: frequencies (U4, U8 - T1)(U8 - T4)(U4, U8 - T8)(U3, U7, U8 - T9)(U4, U8 - T10) such as (S)ARMA (U5, U6 - T5), probability such as fuzzy logic (U5, U7 - T3), and Markov process (U5, U6 - T5)(U3, U6 - T7)(U4, U8 - T8).

(HU3) (Re-)Configuration This group is divided into two parts: configuration at design time such as adding more sensors to increase accuracy (U3 - T3) and reconfiguration at runtime (U3, U6 - T2)(U3, U5 - T3) such as dynamic communication interfaces (U4 - T4)(U1 - T12) or deactivate a malfunctional sensor (U6 - T7).

(HU4) Machine Learning and Neural Networks This group contains unsupervised and supervised learning such as neural network (U1, U3, U5, U6 - T7)(U3, U5 - T6) and SVM (U4 - T1)(U4 - T2)(U3, U5 - T3)(U3, U8 - T9)(U4, U8 - T10), and reinforcement learning (U1, U2, U4, U8 - T4).

(HU5) Verification This group contains design time and runtime verification such as SMC-BIP statistical model checking (U6 - T11).

(HU6) Human-in-the-loop This group contains total involvement from human in the decision such as notify users about their energy consumption (U2 - T2)(U4 - T10) or human giving orders by gestures (U3, U5 - T3), or partial involvement such as robot-human cooperation to perform rescue mission (U2 - T9).

(HU7) Declarative Programming[8] This group contains constraint programming such as ensembles of entities that are formed based on constraints (U4, U6 - T11), modeling such as behavioral trees that mathematically models the execution of tasks (U3, U6 - T6)(U6 - T11), and logic programming such as Prolog that uses facts and rules to infer information (U1 - T12).

We provide a simplified overview on the methods used in each group with their assumptions, inputs and outputs (see Tab. 17).

6 Reasoning: A Guide for Developers

In this section, we discuss the findings of our investigation of the projects concerning the process of developing autonomous components in CPSs that consider uncertainties. We start with general steps (Fig. 6) and then proceed to a more detailed description (Fig. 7). We conclude with a guide for designing such components inspired by aspects of Dependable Systems Design from Software Engineers [20]. The guide is followed by two check lists for the selection of the methods and the self-adaptation choices. The check lists are synthesized from the projects and do not cover all the possible cases. It is rather a reflection of current industrial practices.

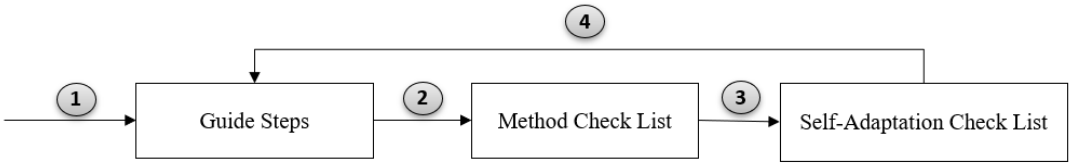


Figure 6: *Overall approach to design uncertainty-aware component in CPS*

Assuming that the scenario is already defined, the overview (Fig. 7) shows the main steps, which starts with identifying the processes, the system properties and the modes for the knowledge structure of the component. These steps include determining the data from sensors and effectors of the CPS components, in addition to considering the data received and submitted to ensure required collaboration among the entities in the system. Later on, the developer should focus on the hazards, if the adaptation is needed that might require mode-switching, considering uncertainty information, methods to handle them, and rules for mode-switching. These details may imply changes to the knowledge structure and thus updating them. It is worth mentioning that this guide does not specify the exact implementation of the autonomous component. It just directs the developers through their design process by asking and considering specific points related to the self-adaptation and uncertainties in the components of CPS.

Simply put, the development process should follow these steps:

1. Complete the steps of the guide in Fig. 25
   (a) Identify the key parts that constitute an autonomous component in a CPS, i.e., processes, properties, inputs, modes, and output types (see Fig. 8).
   (b) Fill the information on the knowledge framework (see Fig. 9).
   (c) Identify the hazards and answer the self-adaptation questions[9] [21].
   (d) Extract the required information mentioned in steps 7,8, and 9 in Fig. 26, which are thresholds*, uncertainty information** and methods to handle uncertainty*** (see Fig. 10, Fig. 12, Fig. 5 and Tab. 16).
   (e) Determine the adaptation decisions (e.g. rules) (see Fig. 14).

[8] We followed the classification mentioned in wikipedia.
<https://en.wikipedia.org/wiki/Declarative_programming>

[9] When to adapt? Why do we have to adapt? Where do we have to implement change? What kind of change is needed? Who has to perform the adaptation? How is the adaptation performed?

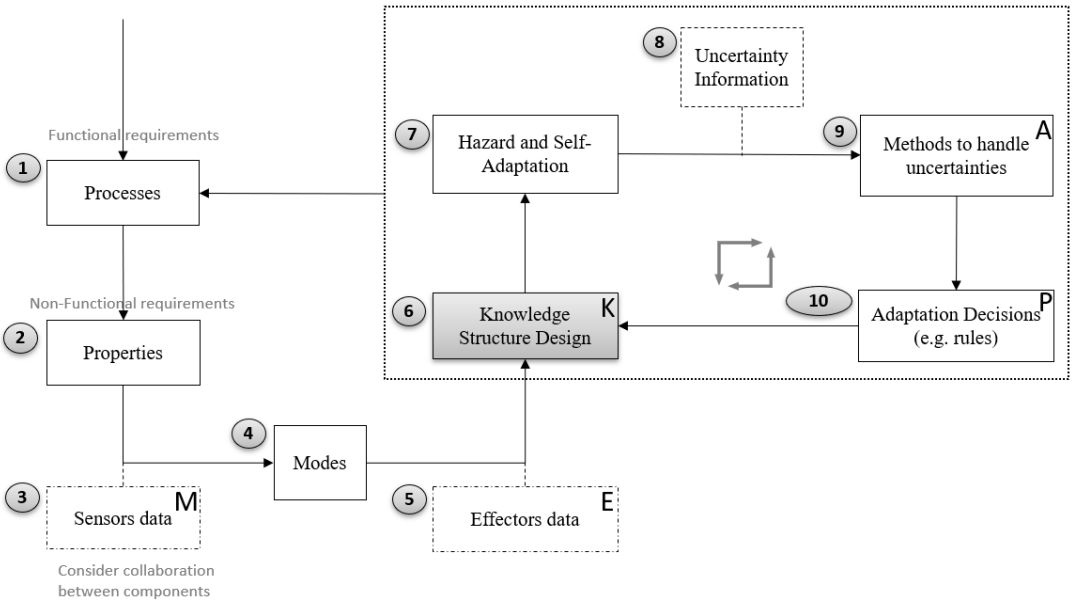


Figure 7: *An overview of a guide for designing uncertainty-aware component in CPS*

(f) Update and fill the information on the knowledge framework (see Fig. 9 and Fig. 11).

2. Run through the methods checklist in Fig. 27 and consider the suggestions extracted from the projects in your design.
3. Run though the self-adaptation checklist in Fig. 28 and consider the suggestions extracted from the projects in your design.
4. Repeat the steps until the design is stabilized.

Component Staring with **the guide steps** in Fig. 25, the developer should identify the *processes* in the component, which form the functional part, then identify the properties, which form the non-functional part. Usually, the system requires a potential output to achieve the property which can be taken into account in the analysis phase (see Fig. 8). We match the system properties to the output types in the projects. For instance, it is interesting to see that the external involvement is used in the projects for addressing performance, resilience and cooperation properties. Having the output types will help us in the final selection of the methods to handle existing uncertainties.

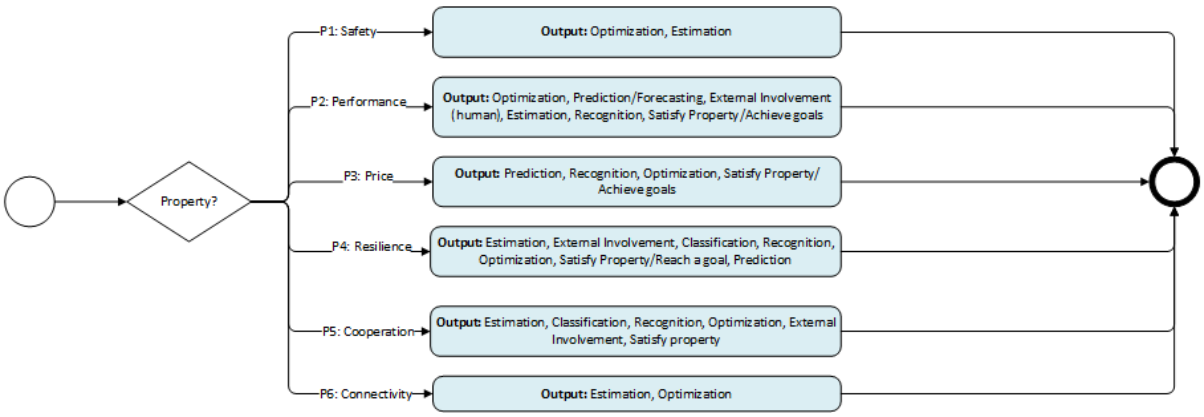


Figure 8: *Properties and Output Types*

It is worth mentioning that the study was focused on the properties only, even though there are other potential factors that can influence the types of uncertainties to consider in the system, such as autonomy level and where the computation is preformed. This will be left for further investigation in the future.

Data Next, the developer should consider different parts of each CPS component including the sensors and actuators and how they impact the data structure inside the component. The data from sensors are captured by corresponding type of data structure that could be defined later, such as variable and time series. Also, the developer needs to consider the inputs received from collaborations with other entities. This step might be unclear or unknown at the first round due to current state of the scenario or the proposed system structure. Therefore, it needs to be rechecked later on when more information is available. In the component, processing the data and handling other functionality are associated to the modes that the component could be in. This step includes defining the relations between the processes and the specified modes. As for actuators, the output of the system is expressed by events or data values that cause the actuators to function in a specific way. At this step, the basic blocks of the component are already specified and it is possible to build the first knowledge structure of the autonomous component. We illustrate the knowledge division on the MAPE-K model in Fig. 9.

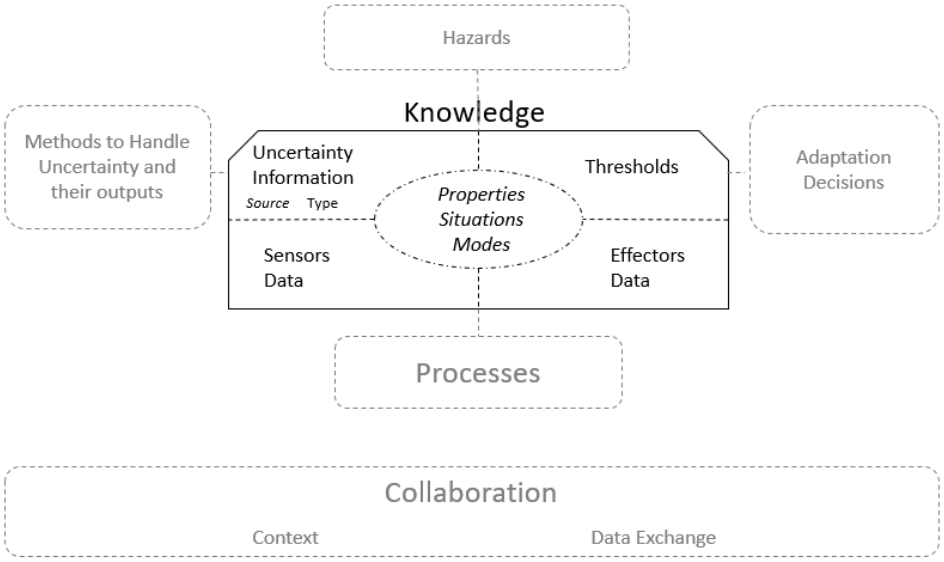


Figure 9: *A knowledge-oriented framework to design uncertainty-aware self-adaptive component in CPS*

Afterwards, the developer should identify the challenges associated to targeted properties in the presented case and hazardous situations associated to them. In our view, the hazards are the root-cause of self-adaptation where manifestation of existing uncertainties might take part in it. Therefore, a more detailed answers related to self-adaptation [21] are to be answered. Then, the developer will be able to identify the sources and types of uncertainties and the appropriate methods to handle them (see Fig. 26).

***Threshold** During identifying the hazards, it is necessary to consider the threshold that determines the switch to hazardous state. The thresholds can be either fixed and captured as a constant in the architecture, or a function of runtime variables. A constant value can be extracted from experiments or estimated. The function of runtime variables can change with the context such as a different constant value for each mode.

****Uncertainty Information** An important aspect in decision making is the uncertainty impact over the functionality in the system. Therefore, the developer needs to identify the uncertainty sources and types in the system, and consider them during the design. The uncertainty sources as they are mentioned before are: model uncertainty, adaptation functions uncertainty, goals uncertainty, environment uncertainty, resources uncertainty, managed system uncertainty. Regarding the types of uncertainty, we classify them as following: network and delays, missing information, noise, peaks, ambiguity and ill-definition, failures, inconsistency, other uncertainties related to context. By identifying the uncertainty types, we can consider possible methods to handle them (see Fig. 10) and use them in the next steps. Moreover, the uncertainty types can be ordered in a tree form to illustrate their structure with regard to data flow (see Fig. 11) starting with the source of uncertainty (i.e. sensors). This structure resemble fault

propagation tree. In our case, it is possible to select methods (i.e. could be represented as a function) to handle each uncertainty type separately or as a group.

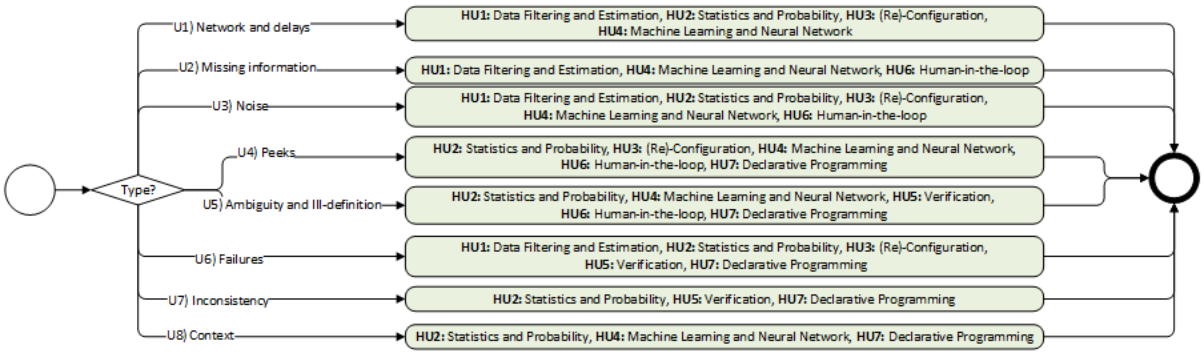


Figure 10: *Uncertainty types and the corresponding methods*

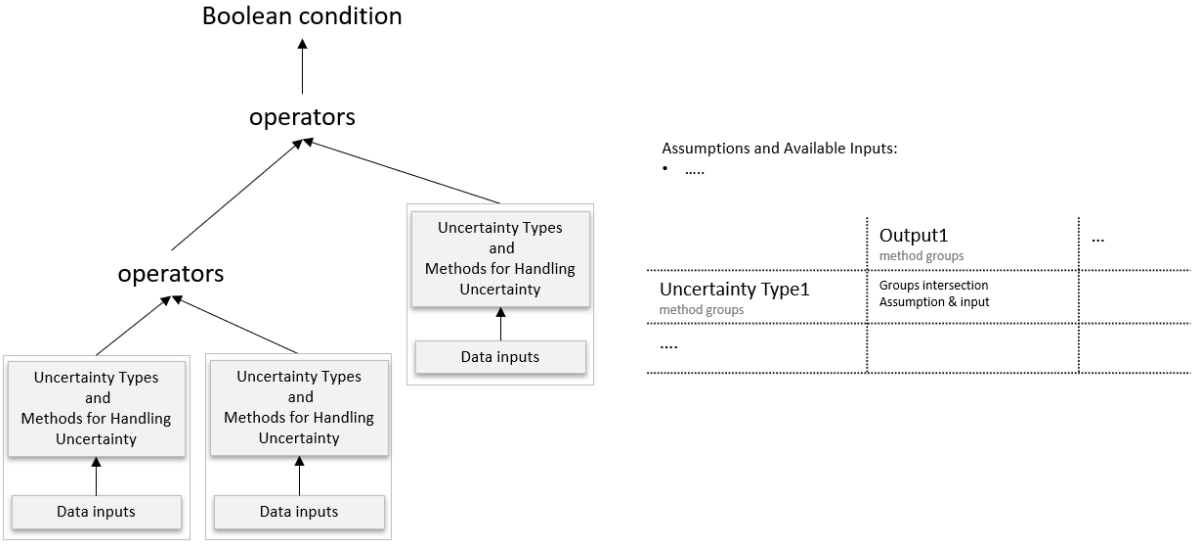


Figure 11: *The selection of the methods to handle uncertainty*

***Methods to Handle Uncertainty** This part considers the important points to ask when choosing suitable methods to handle uncertainties. The details are explained previously and illustrated in Fig. 5. We propose the following order to be used in the selection process after knowing the uncertainty sources and types Fig. 13:

1. Identify the reason of using the methods, which are: avoiding uncertainty, removing uncertainty, mitigating uncertainty, and forecasting uncertainty.
2. Check the output types, assumptions and the input types for each possible group from Fig. 8, Fig. 10 and Fig. 12.
3. Select the potential methods that fit into the determined group. To do so, use Fig. 11 and Tab. 16.
4. Identify the structure of the methods if there are more than one to be used statically or adaptively.
5. Check if all the uncertainty sources and types are covered by the methods. If not, return to step 1.
6. Check if the assumptions in the selected methods are limiting. If yes, return to step 1.

Regarding Tab. 16, the intersection in the methods used to get a specific output types and the methods required for handling uncertainty (i.e. notice that the intersection is the preference but the solution

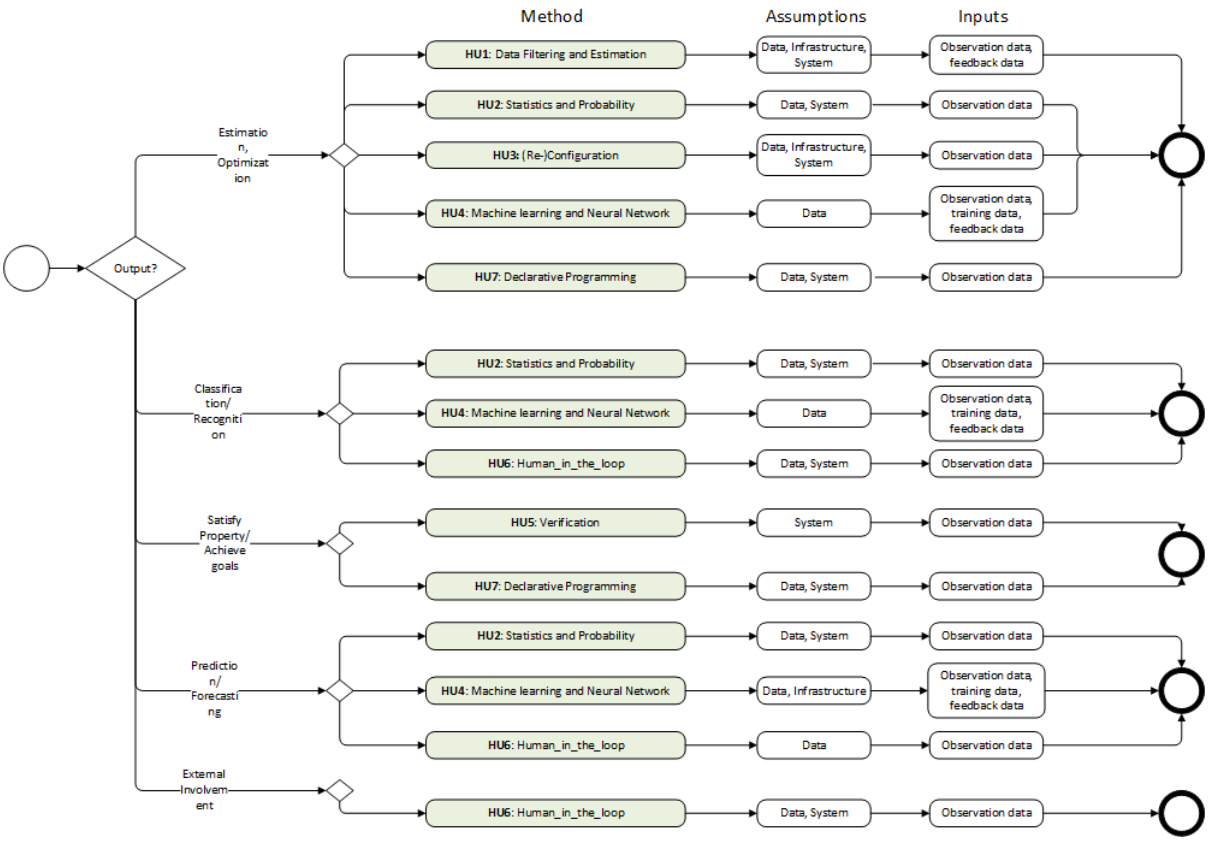


Figure 12: *Output types and corresponding methods with their inputs and assumptions*

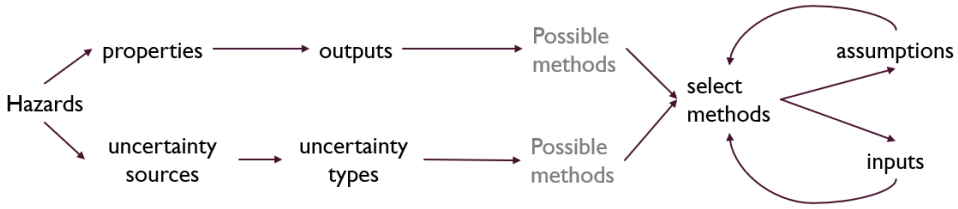


Figure 13: *Simple Steps to select the methods that handles the uncertainty*

	Estimation/ Optimization HU1, HU2, HU3, HU4, HU7	**Classification/ Recognition** HU2, HU4, HU6	**Satisfy Property/ Achieve goals** HU5, HU7	**Prediction/ Forecasting** HU2, HU4, HU6	**External Involvement** HU6
Network and Delays HU1, HU2, HU3, HU4	HU1, HU2, HU3, HU4	HU2, HU4	-	HU2, HU4	-
Missing Information HU1, HU4, HU6	HU1, HU4	HU4, HU6	-	HU4, HU6	HU6
Noise HU1, HU2, HU3, HU4, HU6	HU1, HU2, HU3, HU4	HU2, HU4, HU6	-	HU2, HU4, HU6	HU6
Peeks HU2, HU3, HU4, HU6, HU7	HU2, HU3, HU4, HU7	HU2, HU4, HU6	HU7	HU2, HU4, HU6	HU6
Ambiguity and Ill-definition HU2, HU4, HU5, HU6, HU7	HU2, HU4, HU7	HU2, HU4, HU6	HU5, HU7	HU2, HU4, HU6	HU6
Failures HU1, HU2, HU3, HU5, HU7	HU1, HU2, HU3, HU7	HU2	HU5, HU7	HU2	-
Inconsistency HU2, HU5, HU7	HU2, HU7	HU2	HU5, HU7	HU2	-
Context HU2, HU4, HU7	HU2, HU4, HU7	HU2, HU4	HU7	HU2, HU4	-

Table 16

Corresponding Methods of Outputs and Uncertainty Types.

HU1: Data filtering and estimation, HU2: Statistics and Probability, HU3: (Re-)Configuration, HU4: Machine Learning and Neural Networks, HU5: Verification, HU6: Human-in-the-loop, HU7: Declarative Programming.

might be not limited to it, especially when considering the assumptions and the available inputs). In other words, we recommend to check the methods from the intersection first, and then consider the union. Please, bear in mind that the data is limited to the projects in the study, which they serve as an example not as a rule.

Adaptation decisions The decisions taken here are related to dealing with hazardous situations, which requires mode-switching. We decided to represent mode switching as a rule in the following format: $f(x, y, ..., thresholds) \rightarrow Mode$.

The rule condition (Fig. 14) is defined in the form of $f(x, y, ..., thresholds)$ and its evaluation is a Boolean value (i.e. could be combined using logical operators later as mentioned before), where x, y, ... are the input data, such as sensory information about an observed object/environment that could pass through data analysis; thresholds are the values to be compared with; the evaluation is an operator or a comparison function. If the condition evaluation returns true, it causes activating the mode associated to that condition.

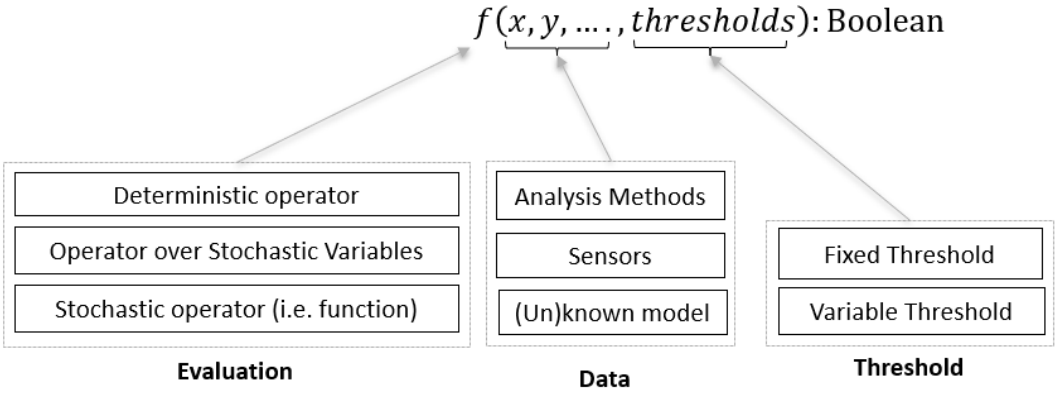


Figure 14: *The rule condition is defined in form $f(x, y, ..., thresholds)$ and its evaluation is boolean value.*

More specifically, the information is gathered from local sensors and from other components about current state of the components themselves or the surrounding environment. The data could be analyzed later on, which requires specific type of inputs depending on the method used such as historical measurements, data points, etc. To operate in specific context, it is important to understand the involved parts in the operating environment that may have known models such as vehicle movement or unknown models such as bacteria growth. Also, the involved hardware must be taken into account. For instance, each sensor has its own precision and uncertainties (e.g. WiFi data has delays and radar data has missing information caused by road curves or hills). Usually, the shortcomings and precision is defined by the industry. The analysis is performed using a selected method to clean and perceive the collected information, where the results will be used in planning and decision making. Nevertheless, the methods themselves also can be error-prone.

The evaluation of the rule condition should be taken into account during selecting the suitable operators: Whether the condition is deterministic (e.g. $>$, $<$, ...), whether the variable is stochastic, and if the function is stochastic.

Methods Check List This check list in Fig. 27 helps the developer to review his/her choices in selecting suitable methods to handle uncertainties. The first point is to consider the reason. During the development, the possible reason is to avoid, remove or forecast uncertainty. In case of avoiding or removing uncertainty, the potential output is configuration or satisfying a property. Forecasting uncertainty requires methods that provide prediction or forecasting. During the system operation, the reasons could be mitigating the uncertainty where the potential used methods should provide estimation, optimization, classification, etc. Also, forecasting uncertainty could be used during the operating phase. The most obvious relation between uncertainty source and the methods is the goal uncertainty where the required output usually is satisfying properties. In that case the possible methods are forming ensembles and verification. The most tangible relation is between the uncertainty type and the potential methods to handle them. We listed the uncertainty types as following: The network and delays are mostly handled by data filtering and estimation, statistics and probability, (re-)configuration, and machine learning and neural networks; the missing information is handled by data filtering and estimation, machine learning and neural networks, and human-in-the-loop; the noise is handled by data filtering and estimation, statistics and probability, (re-)configuration, machine learning and neural network, and human-in-the-loop; the peaks are handled by statistics and probability, (re-)configuration, machine learning and neural network, human-in-the-loop, and declarative programming; ambiguity

and ill-definition are handled by statistics and probability, machine learning and neural networks, verification, human-in-the-loop, and declarative programming; the failures are handled by data filtering and estimation, statistics and probability, (re-)configuration, verification, and declarative programming; inconsistency is handled by statistics and probability, verification, and declarative programming; the context is handled by statistics and probability, machine learning and neural network, and declarative programming. Regarding the structure, the assumptions that the developer should take into account are related to data (e.g. probability distribution, correlation, etc.), infrastructure as for the hardware used (e.g. consider the financial aspect) and communication (e.g. require short range communication), and the system itself (e.g. states, probabilities, etc.). It is important to consider the available hardware and their maintenance comparing to the benefits they can provide even though we did not tackle that issue in the study. Finally, for the input types and the output types, the developer can re-check Fig. 12.

Self-Adaptation Check List This check list (Fig. 28) helps the developer to review his self-adaptation design choices by looking into some derived information from the projects. Starting with the time of adaptation, when considering proactive systems, the used methods in analysis should provide forecasting or prediction. Regarding the reason of adaptation, the developer should make sure that the corresponding data is available. For instance, if the reason of adaptation is to perform changes caused by the user, the data that reflects the user changes must be included. The level of adaptation mostly rise a question of association of the adaptation process with the infrastructure such as in case of processing on edge, cloud or peer-to-peer structure. Nevertheless, this choice is strongly related to the domains considered and the properties they target (see Tab. 14).

The selected adaptation technique proposes changes of parameters, structure or context, which impacts the values of the parameters, the communication among different parts in the structure, and the effectors. As proposed in [21], the answer of who does the adaptation is usually that it is automatic, nevertheless there are some cases that consider involvement of human or even giving him/her the control in specific situations. Therefore, it is important to consider the emotional aspect in critical systems as in Crisis and Emergency cases, and the readiness of the human to take over the control, especially in critical situations. Generally, the system should include a possibility of moving to a safe state. In the adaptation control, we highlighted the aspect of decentralization, in cases that the autonomy degree can change during the adaptation process, the developer should consider the needed information in each autonomy degree.

We do not aim at a general approach. Our methodology relies only on a limited set of solutions in selected projects, and would benefit from a more thorough validation. Nevertheless, it captures the basic concepts that are necessary to design autonomous components with considering uncertainties. Also, the study shows the possible interesting future research that can be done in that area. For instance, a closer look on the inputs and assumptions of the methods could be extended in a more extensive and detailed study.

7 Evaluation

In this section, we illustrate the use of the developers guide to design uncertainty-aware component on cleaner robot example and platoon example.

7.1 Cleaner Robots

The example involves two cleaner robots that are responsible for cleaning two interconnected offices (see Fig. 15[10]) (i.e., one robot for each room). Each cleaner robot has a camera to observe the environment and to detect and locate the dirt on the ground. This makes it possible for the robot to clean the spots totally. The cleaner robot also features a battery sensor. This sensor detects the current level of the battery charge. The cleaning takes place depending on the cleaning schedule and the battery level. However, the next scheduled meeting starts in half an hour and the room needs cleaning which takes around one hour.

The goal in this scenario is to clean the room with considering occupants comfort. A simple cleaner robot design has manual adding for cleaning schedule, and a simple evaluation of the battery level for recharging decision. This means any changes happen on the schedule during the year needs manual resetting. A developer with no expertise in smart buildings might miss the possible use of data from

[10]The Figure was created using https://home.by.me/en/

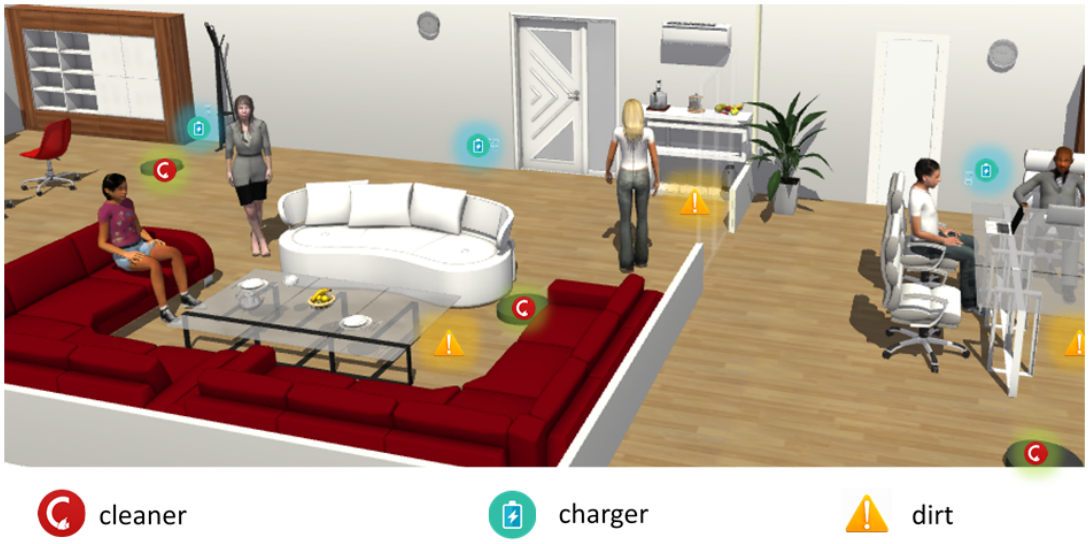


Figure 15: *Cleaner Robots in Offices*

door sensors. The sensors at the door detect passing people through the door. The data collected over days and weeks about the room occupancy allows the cleaner robot to predict the occupancy in the future and set up the cleaning schedule accordingly. Such schedule updates are necessary to consider the ad-hoc events in the offices as well as accommodate schedule modifications.

Another issue is how to deal with the hazardous situations, which are a meeting starts before finishing the cleaning or the battery level is less than 20% before finishing the cleaning. The simple solution is to use the short cleaning program or update the schedule manually. Nevertheless, it is better to consider supporting the robot cleaners collaboration to accomplish the cleaning mission in time. In other words, if the cleaner robot is not able to achieve the job on its own, it asks the second robot in the other room for help. In case the other cleaner robot is available, the cleaning area is divided depending on the available time and battery level, then each robot cleans its assigned part alone. This collaboration provides a better management for time and energy with taking into account the occupants comfort.

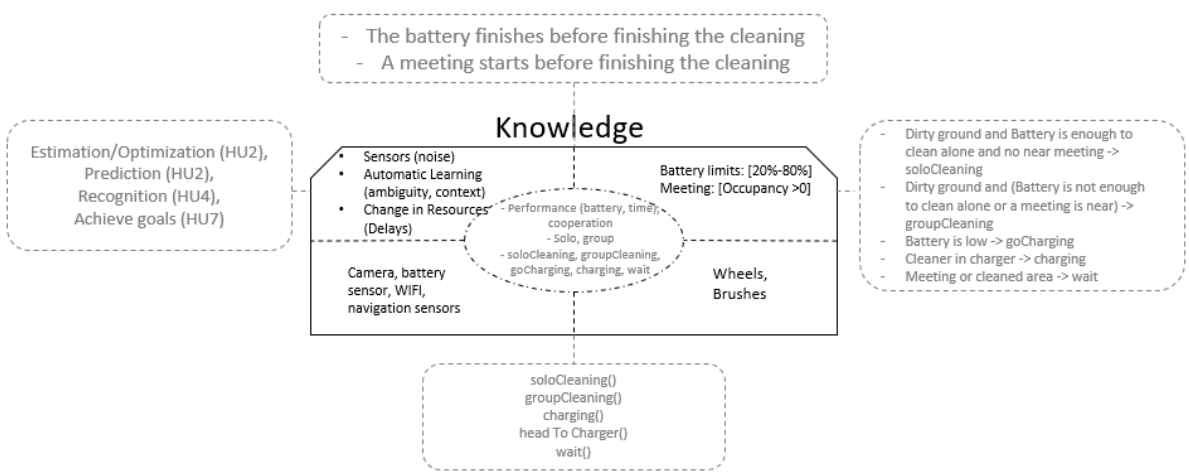
However, the self-adaptation decisions in the cleaner robot are based on evaluation over data with inherent uncertainties. In the aforementioned scenario, operation modes can be employed to simplify management of the robots' activities. Switching between them is driven by the conditions formed over the relevant analysed data. In particular, the cleaning mode is activated when the predicted occupancy of the room is zero for the next one hours and the battery level is above 80%. The uncertainty in this case lies in detecting the dirt, failures in detecting incoming people, communication delays , irregular meetings and erroneous battery-level measurement. All these uncertainties impact the adaptation decisions to some degree, and the designer should foresee them and decide about each particular one as to whether to consider it in the decision process and how to deal with it (i.e. could be learned from other domains).

The evaluation of data relevance and its importance to the scenario is related to the domain and the specification of the use case itself. For instance, each robot cleaner has also navigation sensors to help in moving around the room, but in this scenario the data from them is not relevant to self-adaptation decisions. Moreover, the issue is not only to identify the uncertainty sources, but also the type of uncertainty and the way they should be handled in the self-adaptation decisions.

Now, we look at our motivation example in more detail. We use the guide to help the non-expert developer in his design from higher level to lower level of the autonomous components. As we can see, after applying the guide, we are able to define the most important information related to the self-adaptive component design. The Fig. 16 shows the hazards, properties, uncertainties, and the rules for adaptation. In addition, we are also able to constraints over data exchange between different components.

In Fig. 17, we demonstrate the selection of the methods to handle uncertainties by checking the required output types and the uncertainty types. Also, Fig. 18 illustrates the uncertainties and the operators involved in the self-adaptation rules we defined in Fig. 16. It is worth mentioning that we combined all the rules together for simplicity; however, it is possible to explore each rule separately.

More specifically, we need to design the decisions to be taken at runtime, considering the uncertain-



Entities Communication	Context	Data Exchange
cleaner <-> cleaner	• Cleaner is in groupCleaning mode and both cleaners are available and the sum of their battery levels is enough for cleaning the room.	• Battery level, cleaning area, cleaning time.
cleaner <-> charger	• Charger of the cleaner and Cleaner is in goCharging mode	• Battery level
cleaner <-> camera	• Camera of the Cleaner	• Images of the surrounding to detect dirt
cleaner <-> door sensors	• Door sensors of the room	• Occupant crossing the door

Figure 16: *The cleaner data using the framework*

Assumptions and Available Inputs:

- Input Types: Observation data, Training data (Floor)
- Data: Time series gathered on runtime (Battery Level and Room Occupancy), seasonal (Room Occupancy)
- System: constraints for collaboration

	Estimation/Optimization HU1, HU2, HU3, HU4, HU7	Recognition HU2, HU4, HU6	Achieve goals HU5, HU7	Prediction HU2, HU4, HU6
Noise HU1, HU2, HU3, HU4, HU6	HU1, **HU2**, ~~HU3~~, HU4 historical time series f_1			**HU2**, HU4, ~~HU6~~ historical time series f_1
Peaks HU2, HU3, HU4, HU6, HU7			**HU7** f_4	
Ambiguity and Ill-definition HU2, HU4, HU5, HU6, HU7		HU2, **HU4**, ~~HU6~~ training data f_2		
Context HU2, HU4, HU7				**HU2**, HU4 historical time series f_3
~~Delays~~				

Figure 17: *Uncertainty types considered in the example and selecting the methods to handle them*

ties that may influence reaching the goals, that is, cleaning the rooms without disturbing meetings.

First, we can observe that the delays in communication and data processing do not need to be taken into account, because the type of the information being exchanged is not sensitive to the delays significantly. On contrary, there are several other types of uncertainties that are to be considered.

The first one is precision of the battery level measurement. This is a relatively simple-to-handle issue, because the precision of this measurement is known in advance. Therefore, we just need to be “on the safe side" (by adding a constant to a required battery level) when deciding whether the battery level is sufficient to perform cleaning. In Fig. 17, the intersection set of method groups is between Noise from uncertainty type side, and Estimation/Optimization and Prediction from the output type. The set consists of Data Filtering and Estimation, Statistics and Probability, (Re-)Configuration, Machine Learning and Neural Network and External Involvement. We do not consider External Involvement as an option

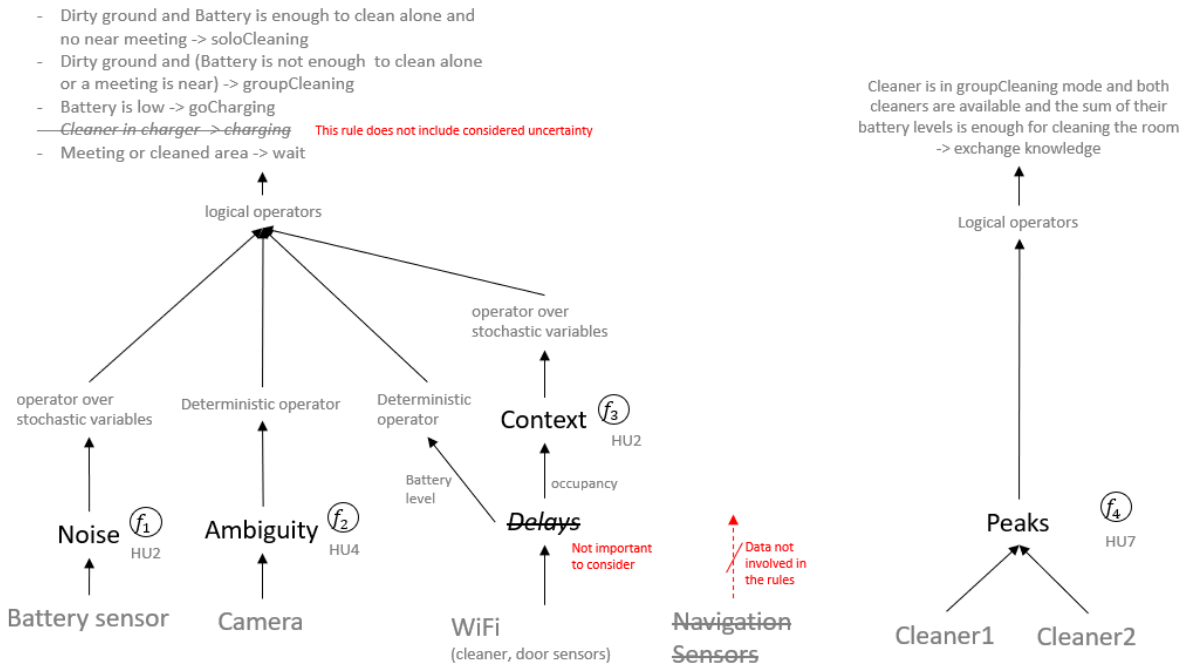


Figure 18: *Uncertainty types and operators in hierarchy to build up the conditions of the rules*

in this scenario, therefore this group is dropped. It is possible to consider Data Filtering and Estimation, but if we consider Kalman Filter for instance that will require knowing the state-space model. The reconfiguration is used in special case such as having other possible sensors that measure the same data (e.g. temperature sensors). This does not apply on battery level measurements, therefore this option is dropped. Regarding Machine Learning and Neural Network, it is possible to use supervised learning for estimation/optimization and prediction, however we do not have training data for battery level as input type. Finally, Statistics and Probability group can be used over the time series and benefit from studying the trends. Here, we select Statistics and Probability group (HU2) by using the method presented in [7], which provides estimation and prediction. However, it assumes that the measurements are independent and identically distributed (i.i.d), and the noise is white.

The second case of uncertainty relates to the camera. It helps in recognizing the dirt on the ground and focus the cleaning on that spot. The decision to take requires some prior knowledge about the floor type and the current environment. In Fig. 17, the intersection set of method groups is between Ambiguity and Ill-definition from uncertainty type side and Recognition from the output type side. The set consists of Statistics and Probability, Machine Learning and Neural Network and External Intervention. As we mentioned before, we do not consider external intervention in this scenario, therefore we drop that option. Looking into the available input types (i.e. training data), the Machine Learning and Neural Network group (HU4) is the best candidate. As a consequence, the robot is trained with images from different types of floor and able to recognize the dirt during the cleaning.

The third case is related to the type of the exchanged data. In our example, this involves the occupancy of the room. In general case, the robot cleaner can be programmed to follow a schedule. However, to make the decision process more accurate in terms of a better prediction, the schedule can be refined with real-time data that is gathered from the door sensors, detecting people entering and leaving the room. In Fig. 17, the intersection set of method groups is between uncertainties related to context from uncertainty type side and prediction from output type side. The set consists of Statistics and Probability, and Machine Learning and Neural Network. Similarly to battery level measurements, it is possible to use machine learning, but that will require training data. The other option is to consider using method from Statistics and Probability group (HU2). The method should be able to deal with time series and seasonality (i.e. meeting schedule) such as ARIMA. This analysis assume that the detection of occupants is not faulty, otherwise it needs to be considered as uncertainty as well.

The fourth case considers balancing the usage of cleaner robot battery and time required to clean. The collaboration is a solution to achieve the cleaning in the case that one of the cleaners required that by switching to group-cleaning mode. This conclusion is derived from the Fig. 17, where the recommended method group is Declarative Programming (HU7). This method group contains a possible technique to

form groups of entities such as ensembles and perform the targeted task.

The involvement of these uncertainties in self-adaptation decision is captured in Fig. 18. It is clear that navigation sensors do not play role in the self-adaptation decision in this scenario, and not all the rules have a defined uncertainty type to handle. It is worth mentioning that we only mention the cleaner $\leftrightarrow$ cleaner communication since it is the only one that addresses one of the uncertainties.

7.2 Platoon Example

Vehicles platooning (i.e. see Fig. 19[11]) or road train is a set of vehicles that has a leader vehicle and a chain of follower vehicles. Each vehicle follows the vehicle in front, while the leader vehicle is responsible for deciding the direction, string stability distance, the speed and changing lanes. Also, the structure of the platoon [15] is dynamic which means vehicles can join and leave the platoon anytime. The basic concern in platoon is keeping string stability. In other words, keeping a constant speed during the ride in the platoon allows vehicles to be driverless and have a minimal safety distance between each other. Of course, the distance is adjustable depending on user preference and due to states such as vehicle merge the platoon or split from it. Consequently, platooning increases the capacity of roads, maximizes safety, minimizes the traffic collisions and congestion, saves time, optimizes fuel consumption, and decreases air drag.

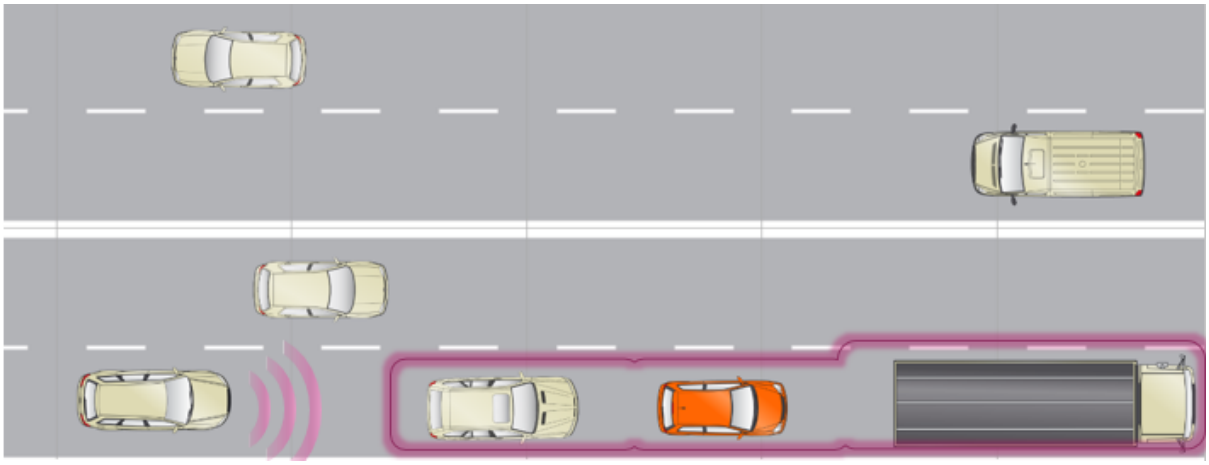


Figure 19: *Vehicle in a Platoon*

Autonomy Degree The modes of the Autonomous Vehicles (AVs) related to autonomy degree (i.e. see Fig. 20[12]) are: Autonomous Warning System (AWS), Adaptive Cruise Control (ACC), and Cooperative Adaptive Cruise Control (CACC). The ACC allows vehicles to use vehicle sensors (i.e. camera, lidar, and radar) to keep a following distance between the vehicles, while CACC uses Vehicle-to-Vehicle (V2V) communication (i.e. Wi-Fi) to improve the ACC. It provides more accurate information and allows for faster response to avoid problems during the rides such as inturns or hilly roads. Nevertheless, switching the situation or autonomy degree from CACC to ACC happens in case of having problems in communication. This might cause early switching that decreases the system efficiency, or fluctuations in switching and disturbance in preserving string stability that leads to collisions. To address this problem, a smooth switching between situations or autonomy degrees within a credible interval ensures the system dependability. This is due to its soft limits in handling the trade-off between situations and their corresponding requirements.

Sensors Autonomous Vehicles (AVs) depend on many sensors to achieve a safe self-driving, which are: Radar (i.e. Radio Detection and Ranging), LiDAR (i.e. Light Detection and Ranging), Cameras, Ultrasonic, GPS (i.e. global-positioning system), accelerometers and altimeters. Generally, radar sensors, lidar sensors and cameras in AV are used in detecting the environment (e.g. pedestrians) since they have relatively long wave ranges. As for ultrasonic sensors, they are more accurate but with shorter

[11] The figure is done using : http://www.accidentsketch.com/

[12] The figure is based on Fig.6 in: "Self-driving cars: The next revolution", KPMG 2012, https://faculty.washington.edu/jbs/itrans/self$_{d}riving_{c}ars$[1].*pdf*

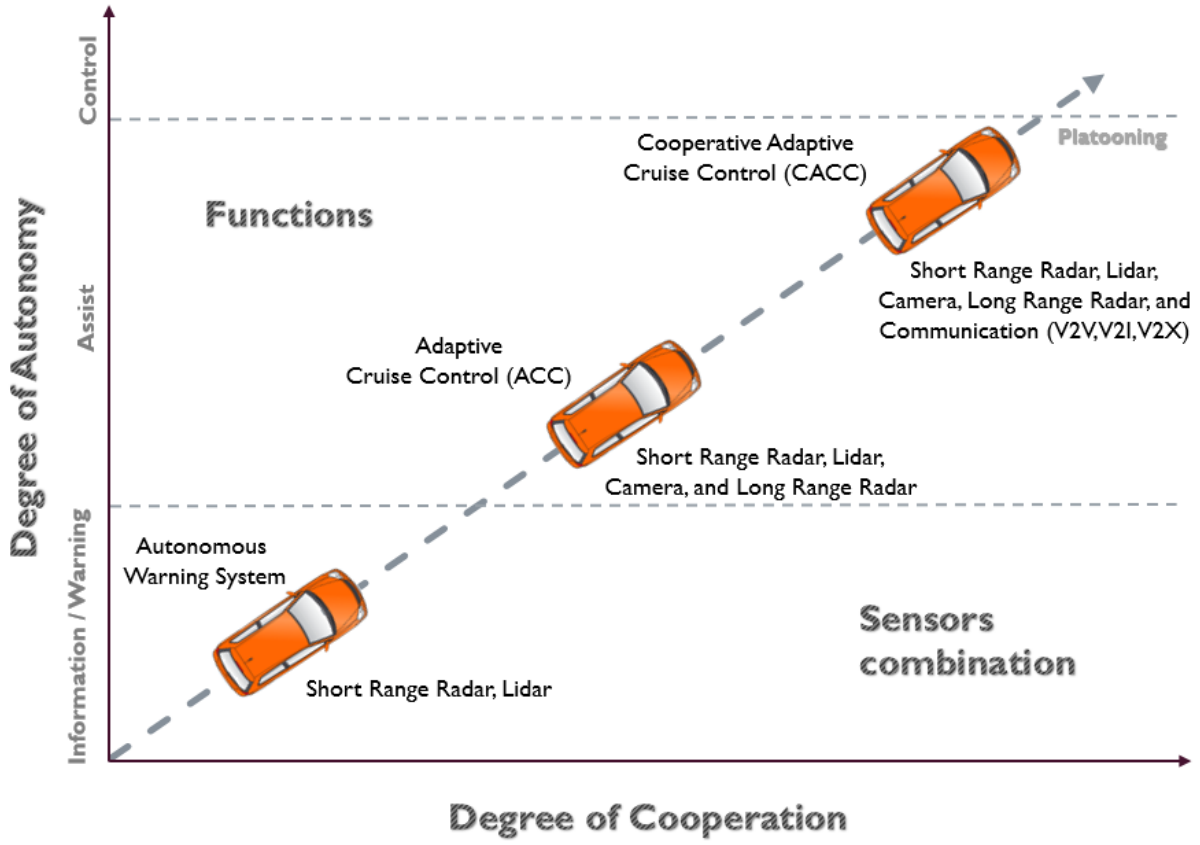


Figure 20: *Autonomy Degree*

range, thus they are involved in parking activity. In our example, we consider four methods of sensing the environment around the vehicle, which are:

- *Radar sensor.* It could be with short 24-GHz or long 77/79-GHz. It has accurate readings and is used to detect both distance and speed of objects around the vehicle. Therefore, it helps in avoiding collisions and parking. Nevertheless, the hilly roads and curves are problematic for radar sensors. As a result, the use of this sensor causes getting measurements with noise and missing information.
- *LiDAR sensor.* It has 360 degree view in addition to accurate reading within 2 cm. It works by sending light pulses and measures the return time of the light to detect the distance. In addition to detecting the distance, it is used also for tracking. As a result, the use of this sensor causes getting measurements with noise.
- *Camera.* It is responsible for detecting things like rear collision alerts, lane departure and pedestrian alert. As a result, the use of cameras causes getting measurements with noise and missing information (e.g. fog).
- *Communication.* The V2V communication is done through Wi-Fi. As a result, many communication problems can impact the quality of the data such as delays and missing information.

Scenario This use case describes four Vehicles in a platoon, which are: Vehicle0 (i.e. The Platoon Leader), Vehicle1, Vehicle2 (i.e. The Reference Vehicle for the Vehicle in interest - the Vehicle in front directly), and Vehicle3 (i.e. The Vehicle of interest which is a Follower). They aim at preserving string stability in the platoon to increase the throughput (i.e. performance). Also, the system cares about safety during the ride, thus the distance between the vehicles should be safe with taking into account the uncertain environment.

The Follower has three ways to get the position of the Leader in front, which are: W1) directly from the reference Vehicle through WiFi, W2) from local sensors, which are: Camera, LiDAR and Radar, or W3) from local sensors, which are: LiDAR and Radar. Each way is related to one mode (i.e. tuple of

concerns). The W1 is used when the vehicles cooperate with each other in the platoon (i.e. CACC), while the W2 is used when the vehicles do not use communication in a platoon (i.e. ACC). Finally, W3 is used when no camera or wifi are involved, only the basic sensors (i.e. short radar and lidar).

The safety distance (i.e. threshold) differs according to the mode. More specifically, the threshold for CACC is half the threshold for ACC and AWS. Although, in all modes the safety distance is a variable that depends on mass and speed of the vehicles in addition to the road nature (e.g. hills), but the scales differ according to the requirements of each mode (e.g. safety distance for CACC is 1.5 m and for ACC or AWS is 5 m).

The Vehicle3 starts with CACC inside a platoon, and depends on the position information from the Vehicle in front itself in addition to the local sensors. The main requirement in this situation is to improve system performance by increasing road capacity, optimizing fuel consumption and decreasing air drag.

Uncertainty and Self-adaptation in the Scenario In this scenario, the uncertainty sources under the study are related to the sensors and the actuators. The measurements from the sensors are assumed to be independent and uncorrelated variables while the modes are dependant and correlated (i.e. they are related to sensors and their corresponding controllers and outputs). Therefore, we need to consider the uncertainty types and the suitable assumptions in each mode and how to handle them.

Some issues related to WiFi data happens, which could be: 1) problems in connection thus there are delays in data exchange. The delays causes a difference between the real location of the Leader and the location that the Follower receives (i.e. position belief); 2) propagation of reference vehicle position through the platoon causes late reaction to a brake (i.e. a lot of fluctuation in the traffic means a higher risk of making accidents). Another issue is the weather condition such as fog and hills, which cause missing information in addition to noise in detecting the reference vehicle. Having both situations make it hard to decide which mode is better to stay in CACC or ACC, or is it better to switch for the AWS as final option. In case of CACC the system focus is on optimizing performance and ensuring safety, while in ACC and AWS is about ensuring safety even if it is on expense of performance.

Design. To design the Autonomous Vehicle component we apply our guide in details as following:

1. Complete the steps of the guide in Fig. 29

   1.1. Identify the key parts that constitute an autonomous component in a CPS, i.e., processes, properties, inputs, modes, and output types.
   Processes. controlAWS, controlACC, controlCACC.
   Properties. safety, performance.
   Situations. safe but focus on optimizing performance (i.e. autonomic), ensure safety first regardless the performance (semi-autonomic).
   Input Types. position of the vehicle in front (wifi, radar, lidar, camera), position of platoon leader (wifi), headway distance (wifi). *Exchange Knowledge.* leader-vehicle (ACC) , V2V (CACC).
   Modes. AWS, ACC, CACC. *Mode Knowledge.* The knowledge differs between the modes because of the difference in the inputs AWS (radar, lidar), ACC (radar, lidar, camera), CACC (radar, lidar, camera, wifi)
   Output Types. Optimal Speed (Gas pedal, brake pedal), Estimation of measurements, Achieve goals of optimizing the usage of roads by being in platoon.

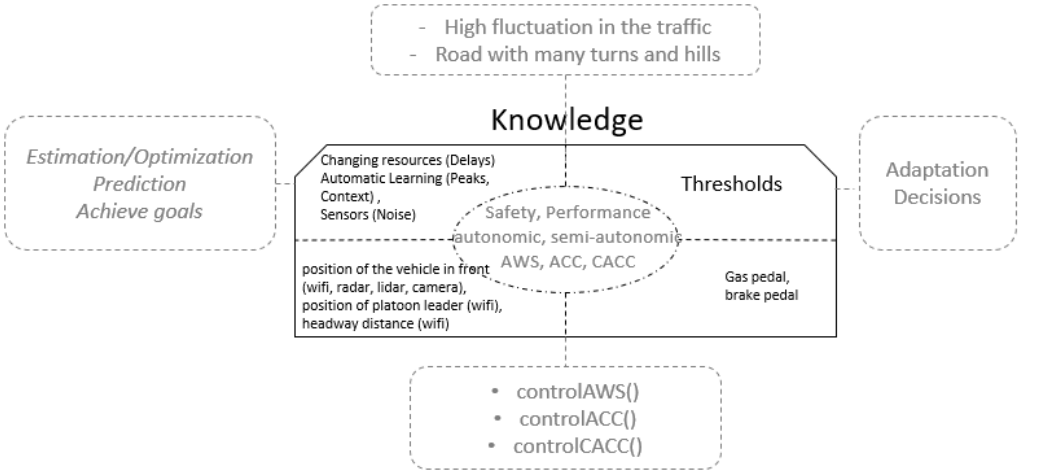
   1.2. Fill the information on the knowledge framework (i.e. see Fig. 21).

   1.3. Identify the hazards and answer the self-adaptation questions [21].
   Identify Hazards.

   - Traffic fluctuations accidents (i.e. while using CACC causes an accident)
   - Having fog, road turns, hills or no wireless communication.

   Answer Self-Adaption Questions.

   - *When to adapt?* Reactive, proactive.
   - *Why do we have to adapt?* Change in the context and technical resources.
   - *Where do we have to implement change?* Ensemble of applications.
   - *What kind of change is needed?* Parameter and structure technique.



Entities Communication	Context	Data Exchange
Vehicle <-> Platoon Leader	The vehicle is In platoon and the mode is CACC or ACC	• Leader position and velocity • Headway distance
Vehicle <-> Vehicle in front	The vehicle is In platoon and the mode is CACC	• Vehicle position and velocity

Figure 21: *The knowledge structure - Initial Extraction*

- *Who has to perform the adaptation?* Human involvement is not required in the control loop.
- *How is the adaptation performed?* Rules, decentralization, external.

1.4. Extract the required information mentioned in steps 7,8, and 9 in Fig. 30, which are thresholds*, uncertainty information** and methods to handle uncertainty***.
Thresholds. Safety Distance (i.e. variable which changes its values for each mode: CACC::1.5m, ACC::5m, AWS::5m.)
Uncertainty Information. *Uncertainty Sources.* Sensors, Changing resources, Automatic Learning *Uncertainty Types.* Noise (i.e. sensor measurements), Network and Delays (i.e. limitations in communication), Peaks (i.e. learn traffic fluctuations). There could be another embedded uncertainty types which might show up during the runtime, nevertheless we do not cover them here.
Method Selection. To start the selection the uncertainty types should be ordered in layers, so we will be able to see the structure of the required methods and the composition of uncertainties. In this example at the first level we consider: noise and delays, while the peaks are on the second level since they are a result of analyzing an estimated data that could be stale of noisy (see Fig. 22). After checking the uncertainty compositions, we should create a table (see Tab. 16) of uncertainty types with their possible method groups, and each needed output types with their possible method groups (see Fig. 12). Also, it is important to state the available input types and assumptions for the use case. First, the selection consider the intersection between the groups for uncertainty types and the output types. Then, we check the possibility of having the existing input types and assumptions with each of the intersected method group (see Fig. 12), in case they do not fit then the selection should be reconsidered from the rest of the method groups (i.e. union of the method groups). During the method group selection a number of questions must be answered:

- *Reasons.* Mitigate uncertainty, Forecasting Uncertainty Impact.
- *Type of Outputs.* Estimation/Optimization, Prediction, Satisfy Property/Achieve goals.
- *Type of Inputs.* Observation data, Feedback data.
- *Potential Method Groups.* HU1, HU2, HU4, HU6, HU7.
- *Structure of Methods.* Multiple methods (adaptive use since the modes include different inputs/sensors).
- *Uncertainty Sources and Types.* Possible to recheck the Adaptation functions uncertainty

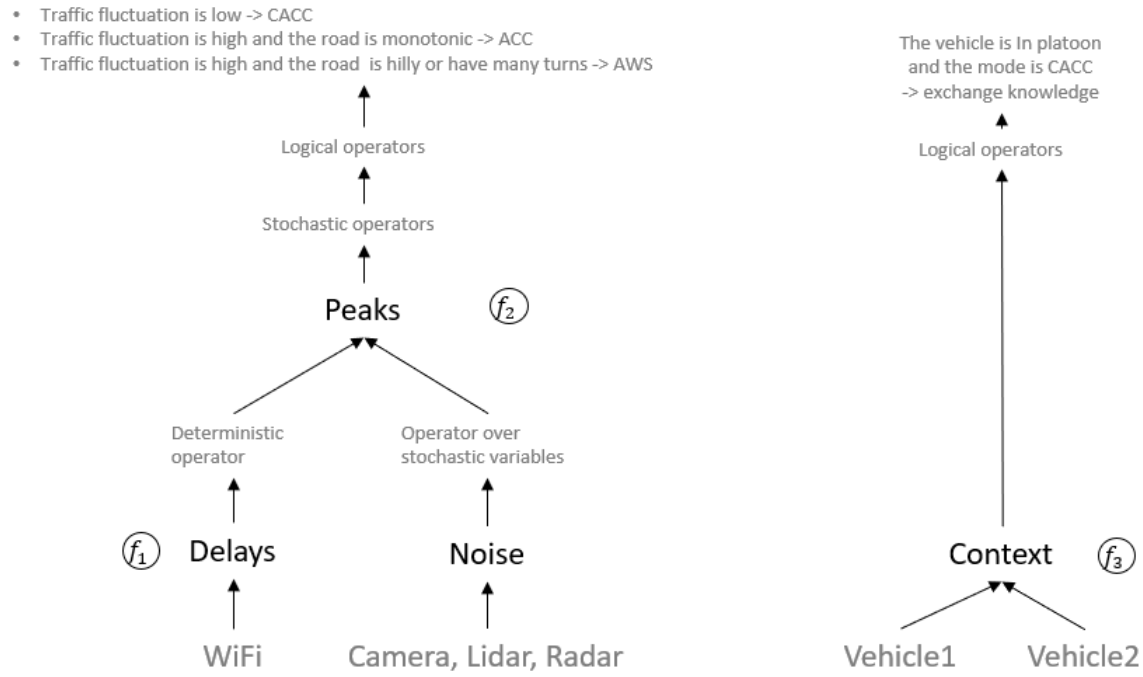


Figure 22: *Uncertainty types composition and Method Selection*

related to Effecting as well, which should be covered at runtime since we have feedback data.

- *Input Types and Assumptions.* Yes, HU4: no available training data. HU7 is used to achieve goals (e.g. ensembles).

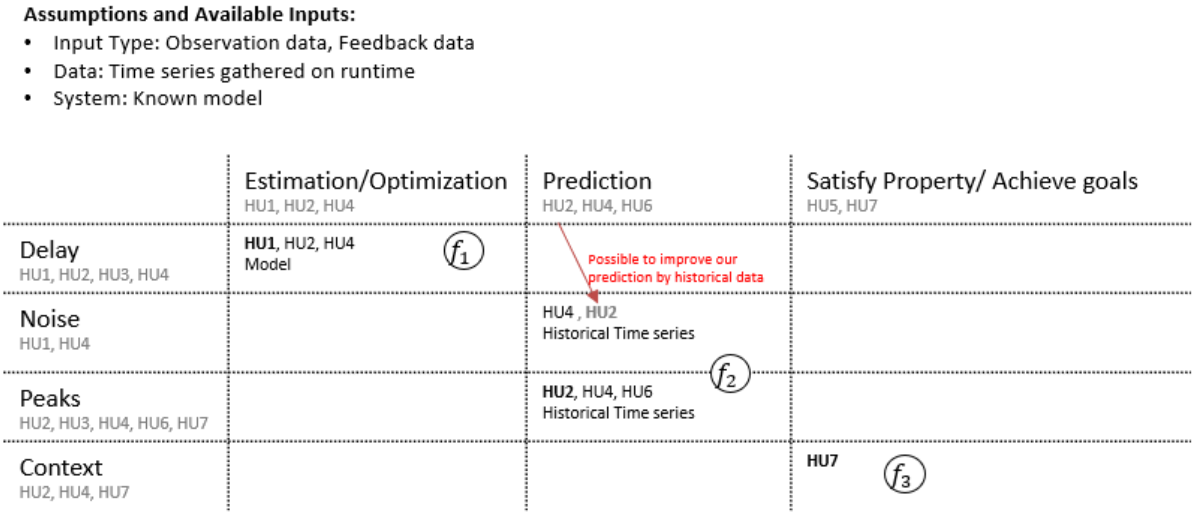
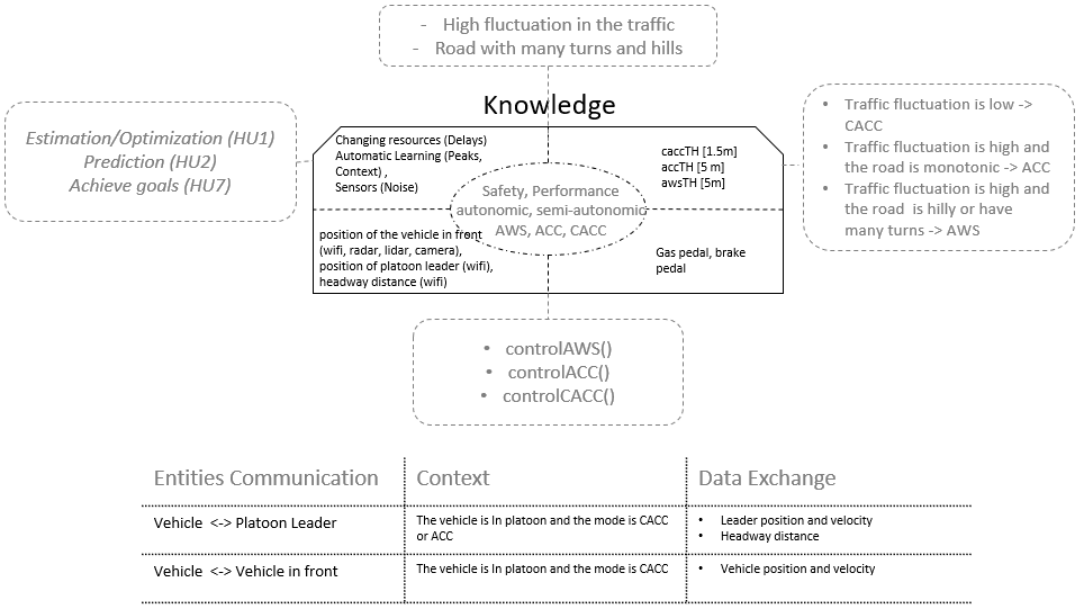


Figure 23: *Uncertainty types and output types table for method selection*

1.5. Determine the adaptation decisions (e.g. rules) (i.e. see Fig. 24)

2. Run through the methods checklist in Fig. 31 and consider the suggestions extracted from the projects in your design.
3. Run though the self-adaptation checklist in Fig. 32 and consider the suggestions extracted from the projects in your design.
4. Repeat the steps until the design is stabilized.



Entities Communication	Context	Data Exchange
Vehicle <-> Platoon Leader	The vehicle is In platoon and the mode is CACC or ACC	• Leader position and velocity • Headway distance
Vehicle <-> Vehicle in front	The vehicle is In platoon and the mode is CACC	• Vehicle position and velocity

Figure 24: *The knowledge structure after method selection*

Listing 1: *Excerpt of the motivating example in DEECo DSL.*

```
component Vehicle
    //Vehicle's role
    knowledge:
          mode,
          situation,
          myposition, myvelocity,
          sensors,    //e.g. leaderPosition, leaderVelocity, ...
          thresholds,
          actuators, ...
     process controlCACC(...)
     process controlCACC(...)
     process controlAWS(...)

     Mode-Switch Board:
     Traffic fluctuation is low -> CACC
     Traffic fluctuation is high and the road is monotonic -> ACC
     Traffic fluctuation is high and the road  is hilly or have many turns -> AWS


ensemble UpdatePlatoonData:
     coordinator: Vehicle as follower
     member: Vehicle as platoonLeader
     membership:
          //The vehicle is In platoon and the mode is CACC or ACC
     knowledge exchange:
          //Leader position and velocity and Headway distance
     scheduling: periodic(200ms)

ensemble UpdateLeaderPositionAndVelocity:
     coordinator: Vehicle as follower
     member: Vehicle as leader
     membership:
          //The vehicle is In platoon and the mode is CACC
     knowledge exchange:
          //Vehicle position and velocity
     scheduling: periodic(200ms)
```

8 Threats to Validity

In this section we discuss the threats to validity of our study. We divide the threats into:

Construct Validity. The first threat is the selection of the primary studies (projects). The SLR guidelines suggest that the primary studies be peer-reviewed papers. But peer-reviewed papers guarantee neither the existence of real industrially-relevant demonstrators, nor do they provide a detailed description of relevant use cases that our study requires. As it is mentioned in [17], the type of research questions in the study determines what is the best selection as primary studies. We therefore chose projects in place of peer-reviewed papers as primary studies. More specifically, we targeted EU projects in ICT calls because projects accepted in these calls are required to deliver real demonstrators (i.e.TRL scale) , and because industrial leadership makes them more likely to be industrially relevant. It is also important to bare in mind that each project has scientific results in addition to a running demonstrator. Using the deliverables, we access both the results of their publications and an overall coherent description of the use cases. Also, we ensure to count the relevant use case just once even though it is mentioned in multiple publications. Although the study is limited to EU projects, there are calls that target collaboration with international parties, which we also considered. It is worth mentioning that the proposed well-known structures of Horizon2020 and FP7 allow us to illustrate how the projects could be used in systematic reviews.

Another threat is related to our selection of project calls. Other calls than ICT might also deliver real demonstrators with industrial orientation. For instance, projects accepted under Innovation in SMEs, Future and Emerging Technologies (FET), or Marie Sklodowska-Curie Actions calls. To mitigate this threat, we did a pre-selection targeting CPS projects in all the calls. The results showed that ICT calls had the largest number of relevant projects.

With respect to the selection criteria, the threat is related to the choice of search keywords. Each criteria could have another set of keywords that the inclusion and exclusion of the projects would depend on. To minimize this threat, we started with much bigger selection of keywords and applied it to 3 calls only. Nevertheless, the results did not show any significant impact on final selection of related projects. We therefore dropped keywords with very broad meaning across different domains, which did not impact the final selection.

Internal Validity. Another threat we should mention is related to domain selection of the use case. There are intersections between the domains and the state-of-the-art lacks for a clear definition of these domains and the boundaries between them. This issue might lead to different results. For this reason, we defined a set of domains that are the most targeted in CPSs and put a supporting examples for each one of them. For instance, having interoperability between different IoT platforms through cloud in which each one targets a different domain (i.e. Smart Traffic and Transport, Smart Homes and Buildings) is under Cloud domain as in case of ClouT project. Another example is robotics as in case of RECONFIG. We chose this domain when the robots are not used in another specific application such as a part of rescue team during crisis.

Conducting Validity. Last threat we would like to mention is conducting the challenges. The problem here is in the level of abstraction the project and use case description have. The high level description of the use case is not enough to determine all the challenges that the use case addresses. To tackle this issue, we defined quality assessment that helps us in rectifying the conducting.

External Validity. We do not claim that the approach is general. It is depends only on limited set of solutions. Nevertheless, it captures the basic concepts that are necessary to design an autonomous component with considering uncertainties. Also, the study shows the possible interesting future research that can be done in this area. For instance, a closer look on the inputs and assumptions of the methods could be extended to more extensive and detailed study (e.g. including learning from feedback data in the statistical and probability group).

9 Related Work

Uncertainty is inherent in CPS, and handling it properly during design and runtime needs sufficient knowledge of multiple domains. Thus, some studies focused on different issues that are related to the problem such as domains and non-functional properties in self-adaptive systems, or building taxonomies for uncertainty types and models and mapping them to self-adaption mechanisms.

Surveys Many studies tried to study different attributes and define taxonomies to relate them to design of self-adaptive systems. For instance, a survey by Villegas et al. [36] deals with mapping system properties in self-adaptive systems to Quality Attributes (QA). The goal is to evaluate a self-adaptive system according to an adaptation metric derived from related quality attributes. This solution helps in avoiding the use of properties that could be unrelated to the adaptation goal (e.g. performance attribute). Malavolta et al. [25] aim at identifying the application domains, challenges, goals, and solutions for architecting CPSs. Their study found links between challenges and the suggested solutions for architecting CPSs. Musil et al. [27] aim at finding out application domains and uncertainty types in self-adaptive CPS, but not mentioning properties. The studies showed interest in connecting the properties of the system with architectural choices, but did not capture the details of uncertainties on behavior.

Adaptation Some frameworks [26, 5] investigate the context that introduces uncertainty in behavior. In [23], the authors present a context taxonomy in addition to a 3-layer framework to design context-aware systems. The authors in [30] aim at reducing the impact of uncertainty in quality evaluation. This is done by defining uncertainty taxonomy and study their sources. The study shows that multiple uncertainties could impact model evaluation. In [32], the study aims at defining taxonomy of uncertainty types, template for their sources, occurrence, and their effect on requirement, design and runtime levels. Even though the studies have multiple shared points with our study, nevertheless they do not propose a clear development process for self-adaptive systems.

As for matching involvement of uncertainty taxonomy in adaptation, in [11], the authors define an uncertainty taxonomy, classify the uncertainty types and match them to MAPE-K stages. However, they do not consider the properties and architectural aspects. To select a suitable uncertainty model, a method is presented in [31]. The authors compare the application of Evidence Theory and Bayesian Theory when handling uncertainty in systems that deal with safety, and provide recommendations for choosing one method over the other in specific cases. Regarding the importance of choosing proper assumptions for input variables, the authors in [6] discuss having different priors for independent inputs and its impact on the posterior probability density distribution. The study proposed probabilistic sensitivity analysis of outputs using log-normal distributions for ratios with uniform prior distribution. In both papers, the focus was on specific class of methods, which is statistics and probability.

Similar to our study, [1] presented a methodology for risk analysis as part of hazard analysis that take into account the security and safety properties. The work consider identifying the uncertainty sources (i.e. parameter, model and completeness uncertainty), uncertainty types and handling them using statistical methods. The proposed framework described four stages starting with collecting input data, characterizing them using fuzzy scale into quantitative and qualitative data, propagating data that is based on risk scenarios (i.e. using combined Bow-Tie and Attack tree to consider both security and safety), and then take decisions on the fuzzy output. Nevertheless, the study is focused on specific properties and using fuzzy logic, while we tried to capture the other used methods in the industry as well.

The [33] discussed uncertainty management by first introducing the uncertainty sources and types then using a framework in shape of uncertainty matrix [34]. The matrix includes the level of uncertainty, its nature and quantification in addition to the models, data and outputs. Nevertheless, it does not capture the properties or how to match it to self-adaptation.

Development Approach Many studies consider linking the requirements to architectural issues. For instance, in [16], the authors target edge computing architecture, where they introduced a classification for computing paradigms within the edge computing (e.g. fog and cloudlet) and listed the solutions presented by such an architecture (i.e. related mostly to network and delays, and performance). Similarly to our approach, they mention the challenges, causes and the guideline to tackle such challenges, even though they did not mention and relation to representation of knowledge as we do.

The study [35] focused on cloud computing and investigated the trends and directions in the next generation, where they associate the layer of abstraction (i.e. operating system, middleware, data, etc.) to changes in the infrastructure, the new computing architecture, and ways of impact (e.g. self-learning systems).

The authors in [9] proposed an architecture for data security that is based on edge-cloud. The architecture consists of 5 layers that reflect the trust level between the organizations. The issue here is to prevent cyber-attacks by data manipulation. This leads to a trade-off between data control and trust in the infrastructure (i.e. fully centralized, hybrid, distributed ISI, Fully distributed). The use case is

specific to the security, but it shows that the architecture is possible to change depending on the requirements.

The paper in [37] proposed a similar approach to ours to develop self-adaptive systems (SAS). It starts by modeling and specifying uncertainties, generating reasoning rules, designing adaptation mechanisms and then decision making and learning knowledge. The schemes to adapt and evolve is based on fuzzy logic and presented through four schemes: forward, backward, parameter-identification and system-identification scheme. Nevertheless, the approach did not consider the specification of different uncertainties and the methods to handle them. Also, the main targeted output is to learn or optimize and adapt accordingly.

To the best of our knowledge, the existing literature does not consider the relation between system concerns (i.e. properties, domains, etc.), types of uncertainty, and uncertainty models as a full chain supported with development guide. Thus, our study adds to the state of the art the knowledge derived from real-life demonstrators about choosing and handling uncertainties. More specifically, it shows if system concerns have an influence on choosing tackled uncertainties and their models. Additionally, we highlight in lesson learned the other possible influences over designing uncertainty-aware self-adaptive component and reason about a guide to help the developers in the process.

10 Conclusion

Dealing with uncertainty in the design of smart cyber-physical systems can be a daunting task. To aid non-expert developers with the aspects of CPS design that deal with uncertainty, we aim to develop a methodology that simplifies the task of identifying the type of a particular uncertainty, as well as identifying an appropriate model and method to incorporate into a CPS to mitigate the effects of thereof.

To this end, we present a systematic review of EU-funded industry-led research projects dealing with collective behavior of smart cyber-physical systems, which identifies links between different high-level aspects of a system and the models and methods used to handle uncertainty relevant to such systems.

Our study is atypical in that we apply the methodology for conducting a systematic literature review to descriptions and outputs of research projects (demonstrators and reports), instead of scientific papers. This choice was made intentionally, since our study focused on challenges and requirements that were relevant for industrial partners who were likely to build such systems, and who had to deliver an actual demonstrator as a deliverable.

The results of our study provide an overview of uncertainty across different domains and challenges, exploring the dependency on domain-specific terminology, context, and requirements. This makes it easier to relate them to CPS designs in different domains, dealing with different challenges using similar uncertainty models and uncertainty handling methods. We also presented a design guide for non-experts that stemmed from the existing information in the projects. This guide provides steps to build the self-adaption rules in CPS components taking into account uncertainty.

References

[1] Houssein Abdo. *Dealing with uncertainty in risk analysis : combining safety and security*. Theses, Université Grenoble Alpes, December 2017.

[2] Rima Al-Ali. Industrial Use Cases of Cyber Physical Systems in EU Projects: Preliminary Study. In *Position Papers of the FedCSIS 2017, Prague, Czech Republic*, volume 12 of *FedCSIS '17*, pages 187–193. Polish Information Processing Society, 2017.

[3] Rima Al-Ali. Description of EU Projects Demonstrate Self-Adaptive CPSs (V1.0). Technical Report D3S-TR-2018-03, Department of Distributed and Dependable Systems, Charles University, 2018.

[4] Samira Bairamzadeh et al. Modelling different types of uncertainty in biofuel supply network design and planning: A robust optimization approach. *Renewable Energy*, 116:500 – 517, 2018.

[5] Gino Baudry et al. Range-based multi-actor multi-criteria analysis: A combined method of multi-actor multi-criteria analysis and monte carlo simulation to support participatory decision making under uncertainty. *European Journal of Operational Research*, 264(1):257 – 269, 2018.

[6] Hendriek C. Boshuizen and Pieter H.M. van Baal. Probabilistic sensitivity analysis: Be a bayesian. *Value in Health*, 12(8):1210 – 1214, 2009.

[7] T. Bures, P. Hnetynka, J. Kofron, Rima Al-Ali, and D. Skoda. Statistical Approach to Architecture Modes in Smart Cyber Physical Systems. In *2016 13th Working IEEE/IFIP Conference on Software Architecture (WICSA)*, pages 168–177, 2016.

[8] Javier Camara, David Garlan, Won Gu Kang, Wenxin Peng, and Bradley Schmerl. Uncertainty in self-adaptive systems: Categories, management, and perspectives, 2017.

[9] David W Chadwick, Wenjun Fan, Gianpiero Constantino, Rogerio de Lemos, Francesco Di Cerbo, Ian Herwono, Mirko Manea, Paolo Mori, Ali Sajjad, and Xiao-Si Wang. A cloud-edge based data security architecture for sharing and analysing cyber threat information. *Future Generation Computer Systems*, 102:710 – 722, 2020.

[10] Rahul Davis. Application of taguchi-based design of experiments for industrial chemical processes. In *Statistical Approaches With Emphasis on Design of Experiments Applied to Chemical Processes*, pages 137–155. InTech, 2018.

[11] Naeem Esfahani and Sam Malek. *Uncertainty in Self-Adaptive Software Systems*, pages 214–238. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[12] Frank Felden, Thomas Kruger, and Wim De Meyer. Tebit 2017 executive report: Time to double down on ai and robotics, 2017.

[13] Volkan Gunes, Steffen Peter, Tony Givargis, and Frank Vahid. A survey on concepts, applications, and challenges in cyber-physical systems. *KSII Transactions on Internet and Information Systems*, 8:4242–4268, 12 2014.

[14] S. Idowu, C. Ahlund, and O. Schelen. Machine learning in district heating system energy optimization. In *2014 IEEE International Conference on Pervasive Computing and Communication Workshops (PERCOM WORKSHOPS)*, pages 224–227, March 2014.

[15] D. Jia, K. Lu, J. Wang, X. Zhang, and X. Shen. A survey on platoon-based vehicular cyber-physical systems. *IEEE Communications Surveys Tutorials*, 18(1):263–284, Firstquarter 2016.

[16] Wazir Zada Khan, Ejaz Ahmed, Saqib Hakak, Ibrar Yaqoob, and Arif Ahmed. Edge computing: A survey. *Future Generation Computer Systems*, 97:219 – 235, 2019.

[17] B. Kitchenham, P. Brereton, M. Turner, M. Niazi, S. Linkman, R. Pretorius, and D. Budgen. The impact of limited search procedures for systematic literature reviews - a participant-observer case study. In *2009 3rd International Symposium on Empirical Software Engineering and Measurement*, pages 336–345, Oct 2009.

[18] B. Kitchenham and S Charters. Guidelines for performing systematic literature reviews in software engineering, 2007.

[19] John Knight. *Fundamentals of Dependable Computing for Software Engineers*. Chapman & Hall/CRC, 1st edition, 2012.

[20] John Knight. *Fundamentals of Dependable Computing for Software Engineers*. CRC Press, 2012.

[21] Christian Krupitzer, Felix Maximilian Roth, Sebastian VanSyckel, Gregor Schiele, and Christian Becker. A survey on engineering approaches for self-adaptive systems. *Pervasive and Mobile Computing*, 17:184 – 206, 2015. 10 years of Pervasive Computing' In Honor of Chatschik Bisdikian.

[22] E. A. Lee. Cyber physical systems: Design challenges. In *2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*, pages 363–369, May 2008.

[23] Yang Lu. Industry 4.0: A survey on technologies, applications and open research issues. *Journal of Industrial Information Integration*, 6:1 – 10, 2017.

[24] Sara Mahdavi-Hezavehi, Paris Avgeriou, and Danny Weyns. A classification framework of uncertainty in architecture-based self-adaptive systems with multiple quality requirements. In *Managing Trade-offs in Adaptable Software Architectures :*, pages 45–78. 1 edition, 2016.

[25] Ivano Malavolta et al. A preliminary study on architecting cyber-physical systems. In *Proceedings of the 2015 European Conference on Software Architecture Workshops*, ECSAW '15, pages 20:1–20:6. ACM, 2015.

[26] Sharif Mohammad and Alesheikh Ali Asghar. Context aware movement analytics: implications, taxonomy, and design framework. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(1):e1233.

[27] Angelika Musil et al. *Patterns for Self-Adaptation in Cyber-Physical Systems*, pages 331–368. Springer International Publishing, Cham, 2017.

[28] V. Pandey et al. Addressing limitations of pareto front in design under uncertainty. In *37th Design Automation Conference, Parts A and B*, pages 1099–1113. ASME, 2011.

[29] Amir Parnianifard et al. An overview on robust design hybrid metamodeling: Advanced methodology in process optimization under uncertainty. *International Journal of Industrial Engineering Computations*, 9(1):1–32, 2018.

[30] Diego Perez-Palacin and Raffaela Mirandola. Uncertainties in the modeling of self-adaptive systems: A taxonomy and an example of availability evaluation. In *Proceedings of the 5th ACM/SPEC International Conference on Performance Engineering*, ICPE '14, pages 3–14, New York, NY, USA, 2014. ACM.

[31] Tomáš Pop et al. Comparison of component frameworks for real-time embedded systems. *Knowledge and Information Systems*, 40(1):127–170, Jul 2014.

[32] Andres J. Ramirez, Adam C. Jensen, and Betty H. C. Cheng. A taxonomy of uncertainty for dynamically adaptive systems. In *Proceedings of the 7th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, SEAMS '12, pages 99–108, Piscataway, NJ, USA, 2012. IEEE Press.

[33] Roger Strand and Deborah Oughton. Risk and uncertainty as a research ethics challenge, 2009.

[34] J.P. van der Sluijs. *Uncertainty, assumptions, and value commitments in the knowledge-base of complex environmental problems*, pages 67–84. Green Leaf Publishing, 2006.

[35] Blesson Varghese and Rajkumar Buyya. Next generation cloud computing: New trends and research directions. *Future Generation Computer Systems*, 79:849 – 861, 2018.

[36] Norha M. Villegas, Hausi A. Müller, Gabriel Tamura, Laurence Duchien, and Rubby Casallas. A framework for evaluating quality-driven self-adaptive software systems. In *Proceedings of the 6th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, SEAMS '11, pages 80–89. ACM, 2011.

[37] Zhuoqun Yang, Zhi Jin, and Zhi Li. Modeling uncertainty and evolving self-adaptive software: A fuzzy theory based requirements engineering approach. *CoRR*, abs/1704.00873, 2017.

[38] Man Zhang, Bran Selic, Shaukat Ali, Tao Yue, Oscar Okariz, and Roland Norgren. Understanding uncertainty in cyber-physical systems: A conceptual model. In Andrzej Wąsowski and Henrik Lönn, editors, *Modelling Foundations and Applications*, pages 247–264, Cham, 2016. Springer International Publishing.

A Method Selection Questions

Project	Reason	Uncertainty Source	Uncertainty Type	Structure	Assumptions	Inputs	Outputs
VICINITY	Mitigate Uncertainty, Forecast Uncertainty	Statistics and Probability, Machine learning and Neural Network, Human-in-the-loop	U4, U8	Multiple-Methods (static use)	Data	observation data	Optimization, Prediction, Extrenal Involvement, Recognition
symbIoT	Mitigate Uncertainty, Forecast Uncertainty	Data Filtering and Estimation, Machine learning and Neural Network	U2, U3, U4, U6	Multiple-Methods (static use)	Infrastructure, System	observation data	Estimation, Prediction, External Involvement
CADDY	Mitigate Uncertainty	Data Filtering and Estimation,Re-Configuration, Machine Learning and Neural Network, Human-in-the-loop, Declarative Programming, Verification	U3, U5, U7	Multiple-Methods (Adaptive use)	System, Data	observation data, training data, feedback data	Estimation, Optimization, Classification, Recognition, Satisfy Property/Achieve goals
OrPHEuS	Mitigate Uncertainty, Forecast Uncertainty	Statistics and Probability, Machine learning and Neural Network, Re-Configuration	U1, U2, U4, U8	Multiple-Methods (Adaptive use)	Data, Infrastructure	observation data, training data, feedback data	Forecast /Prediction, Optimization
ClouT	Mitigate uncertainty	Statistics and Probability	U5, U6	Multiple-Methods (Adaptive use)	Data, System	observation data	Classification
RECONFIG	Mitigate uncertainty	Declarative Programming, Statistics and Probability, Machine Learning and Neural Network, Data Filtering and Estimation	U1, U3, U5, U6	Multiple-Methods (static use)	Data, System	training data, observation data	Satisfy Property/Reach a goal, Classification, Estimation, Recognition
HYDROBIONETS	Avoiding uncertainty, Mitigating uncertainty, Forecast Uncertainty	Machine learning and Neural Network, Data Filtering and Estimation, Re-Configuration, Statistics and Probability	U1, U2, U3, U5, U6	Multiple-Methods (Adaptive use)	Data, System, Infrastructure	training data, observation data	Prediction, Estimation, Optimization
INERTIA	Forecasting uncertainty	Statistic and Probability	U4, U7, U8	Multiple-Methods (static use)	Data, System	observation data	Recognition, Optimization, Prediction
NiFTi	Mitigate uncertainty	Statistics and Probability, Data Filtering and Estimation, Machine Learning and Neural Network, Human-in-the-loop	U1, U2, U3, U7, U8	Multiple-Methods (static use)	Data, System	training data, observation data, feedback data	Classification, Estimation, Optimization, External Involvement
GreenCom	Forecasting uncertainty	Statistics and Probability, Machine learning and Neural Network, Human-in-the-loop	U4, U8	Multiple-Methods (Adaptive use)	Data	training data, observation data	Prediction
ASCENS	Mitigate uncertainty, Forecasting uncertainty	Statistics and Probability, Declarative Programming, Verification	U4, U6, U8	Multiple-Methods (static use)	Data, System	observation data	Satisfy Property/Achieve goals, Prediction, Optimization
IPAC	Mitigate uncertainty	Statistics and Probability, Re-configuration, Data filtering and Estimation, Declarative Programming	U1, U2, U3, U7	Multiple-Methods (static use)	Data, Infrastructure	observation data	Optimization, Estimation

Table 17
Answering Method Selection Questions

B Developer's Guide Figures

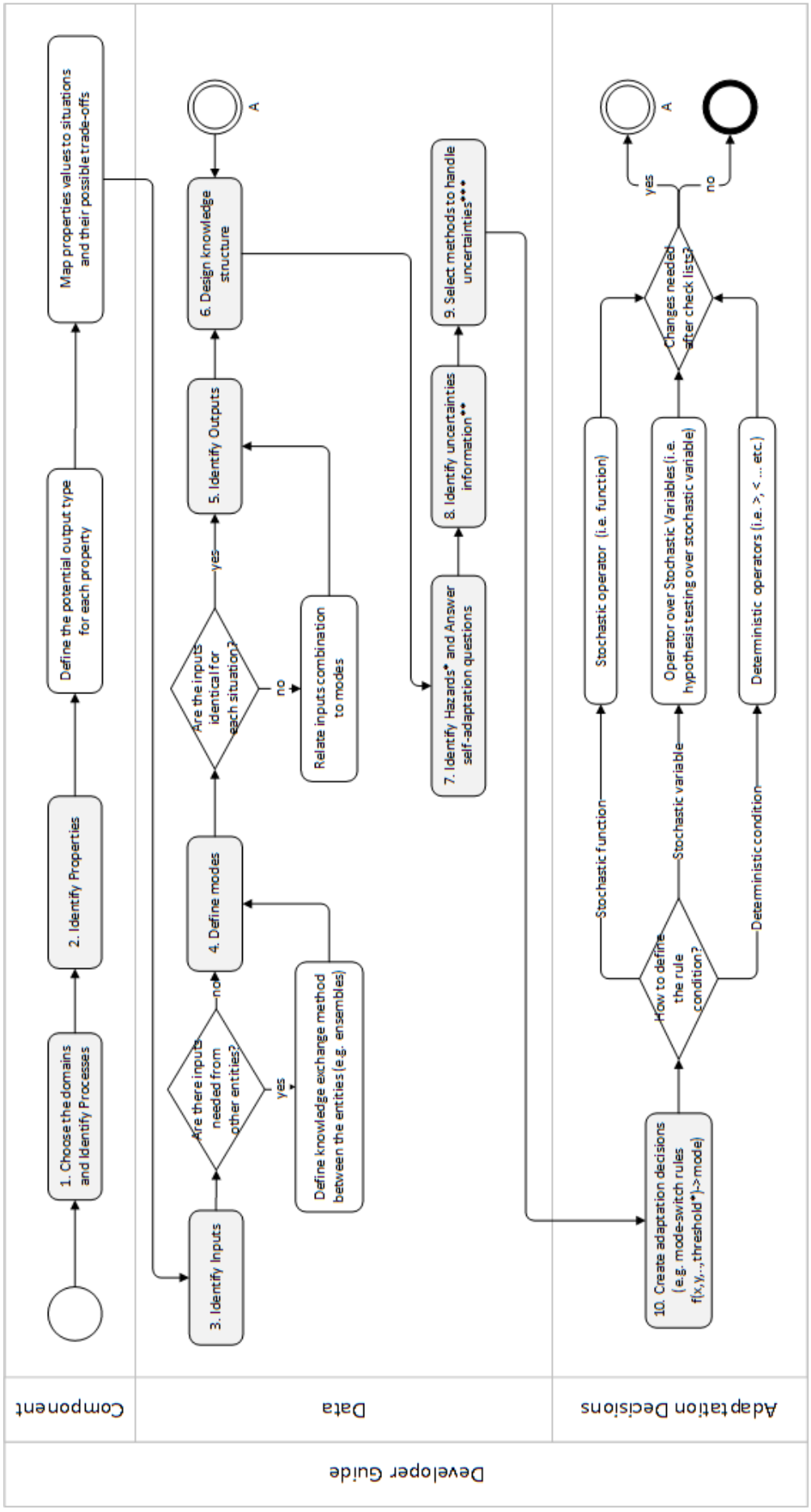


Figure 25: *Developer's Guide*

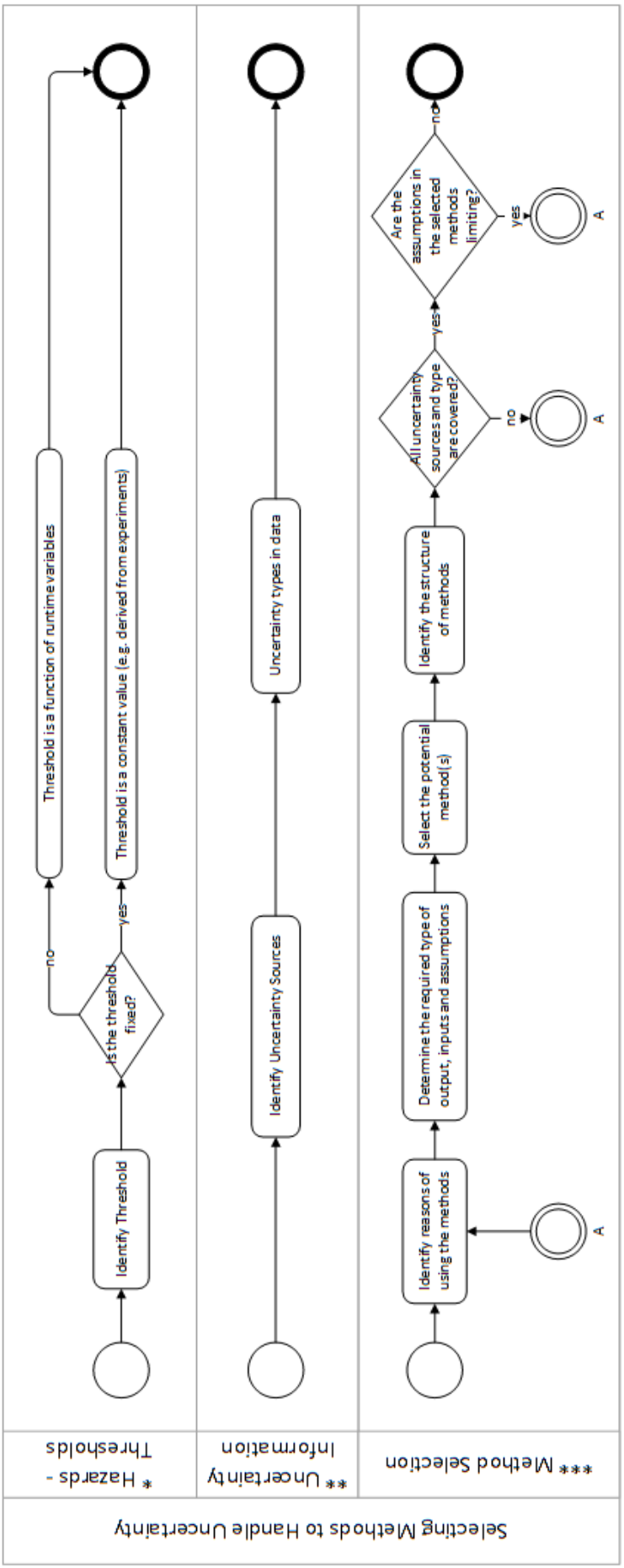


Figure 26: *Detailed guide for developers to consider uncertainty in their design*

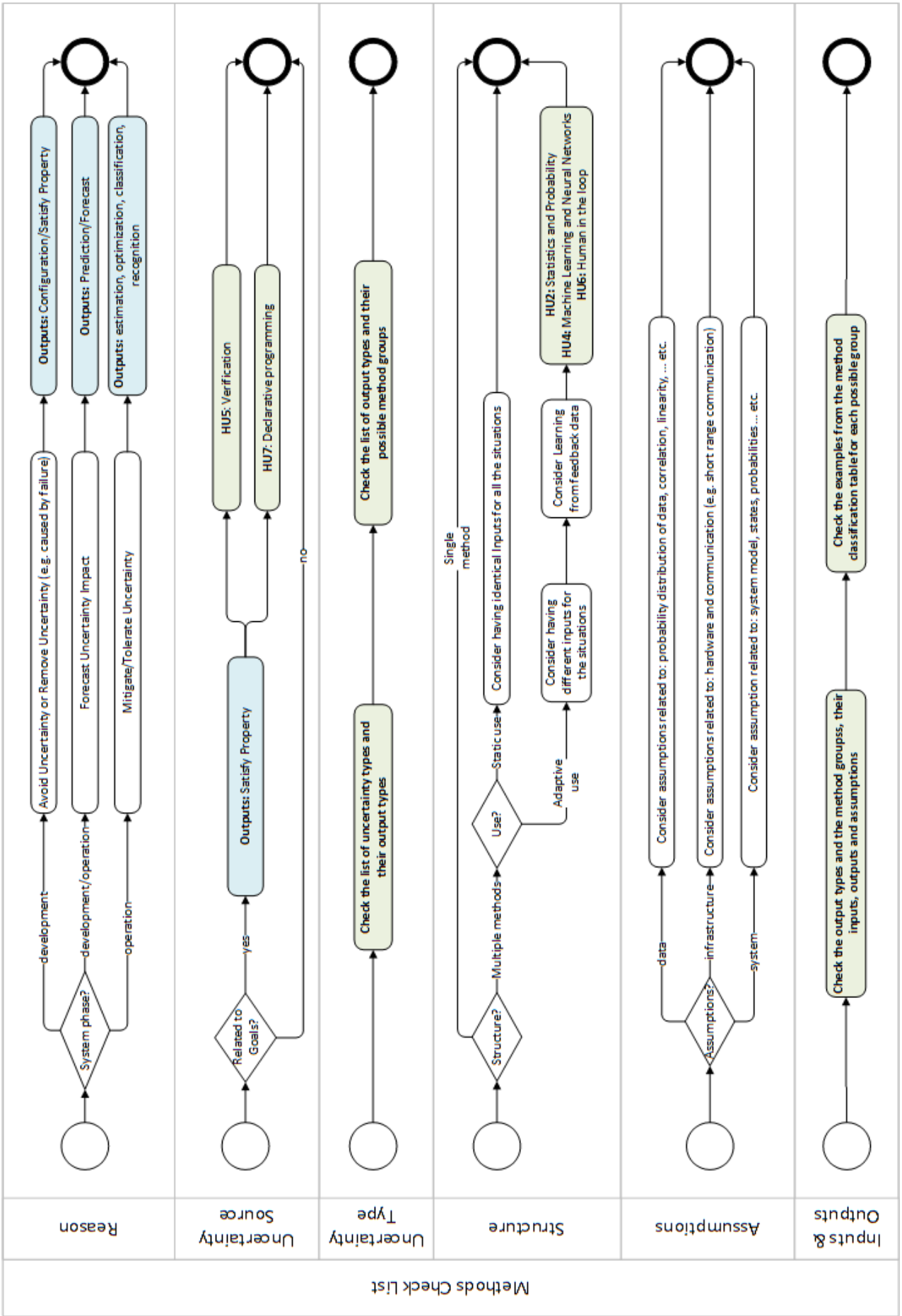


Figure 27: *Checklist over the selected methods inferred from the projects*

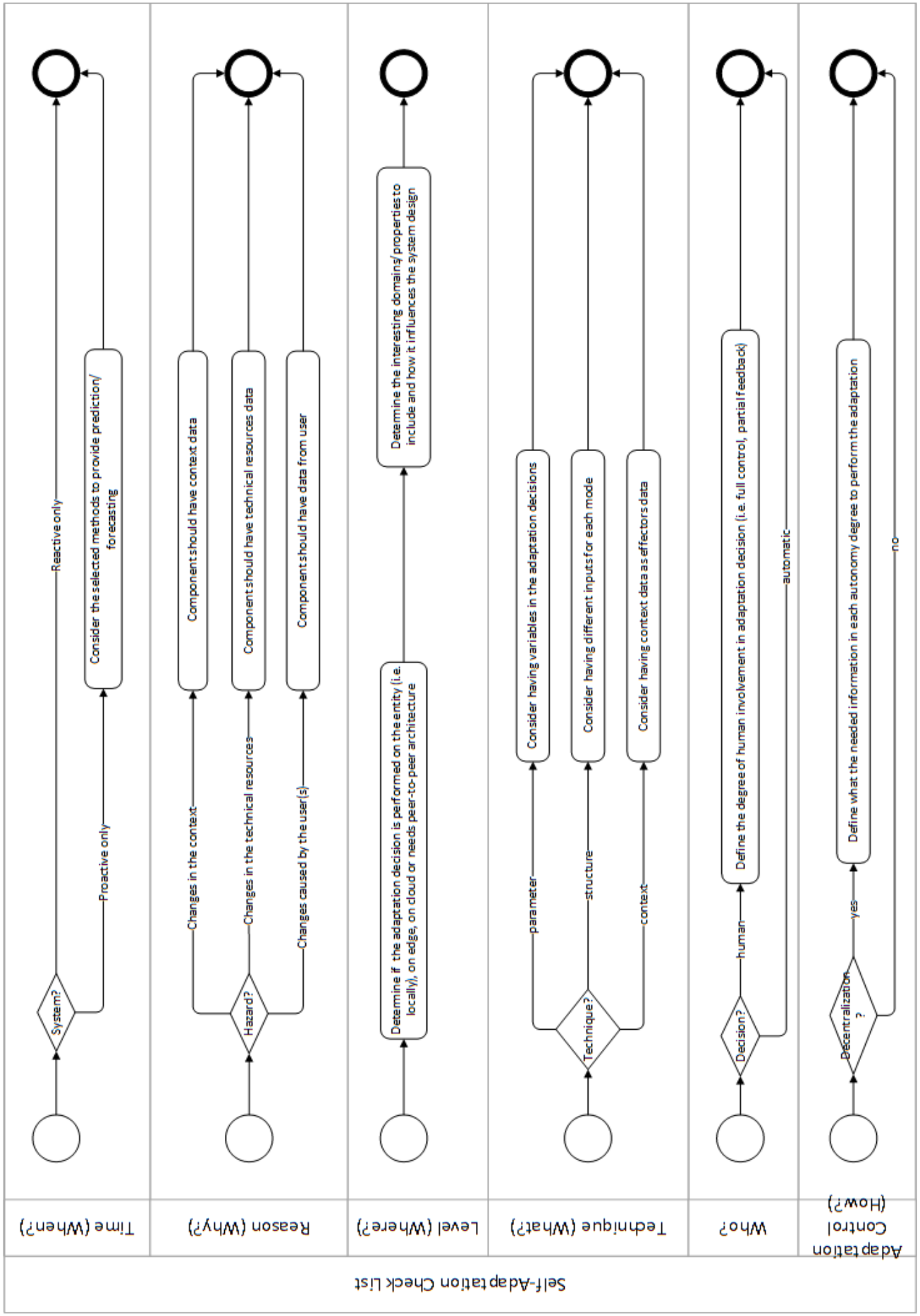


Figure 28: *Checklist over self-adaptation questions inferred from the projects*

C Platoon Example Figures

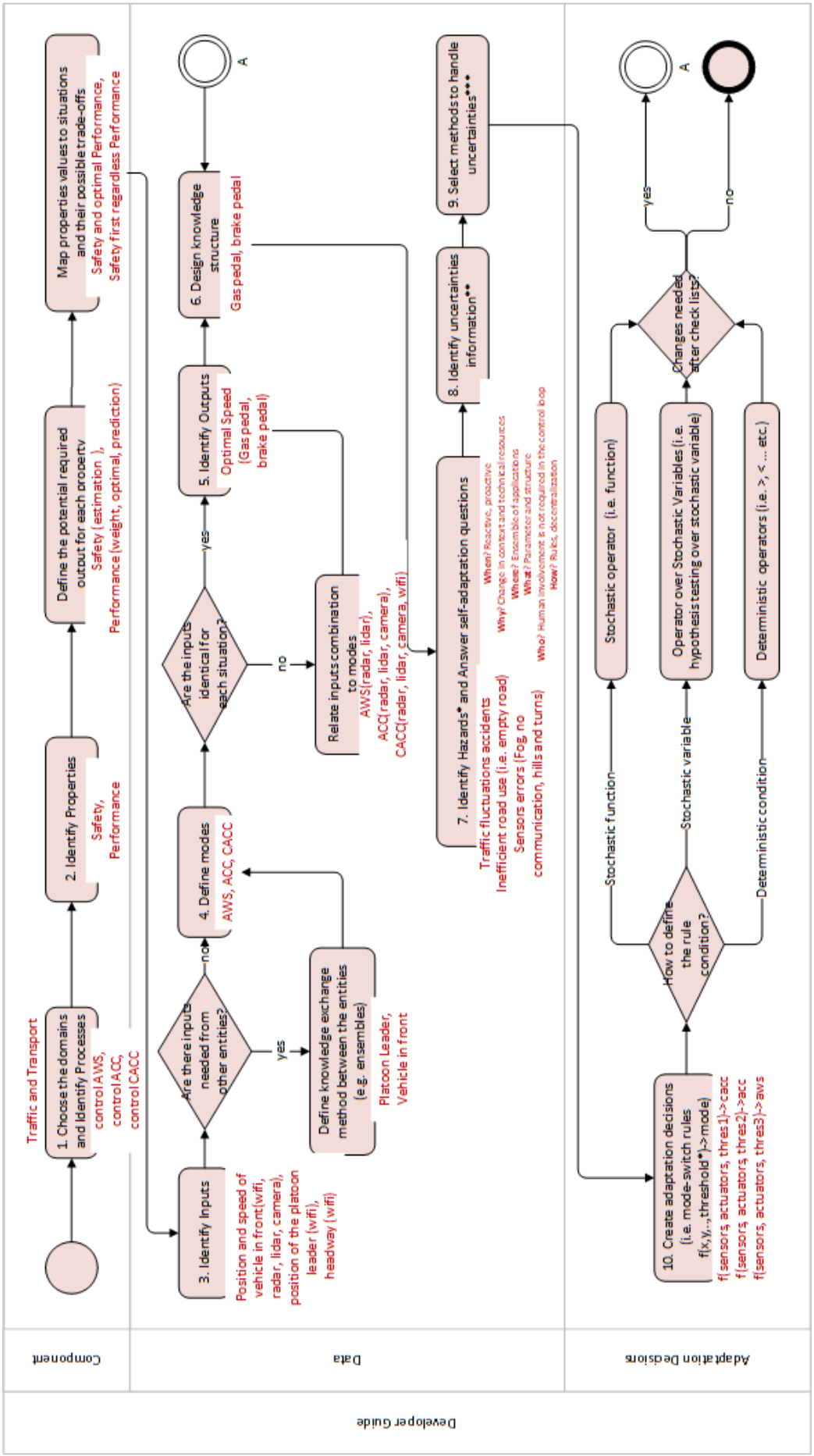


Figure 29: *Developer's Guide*

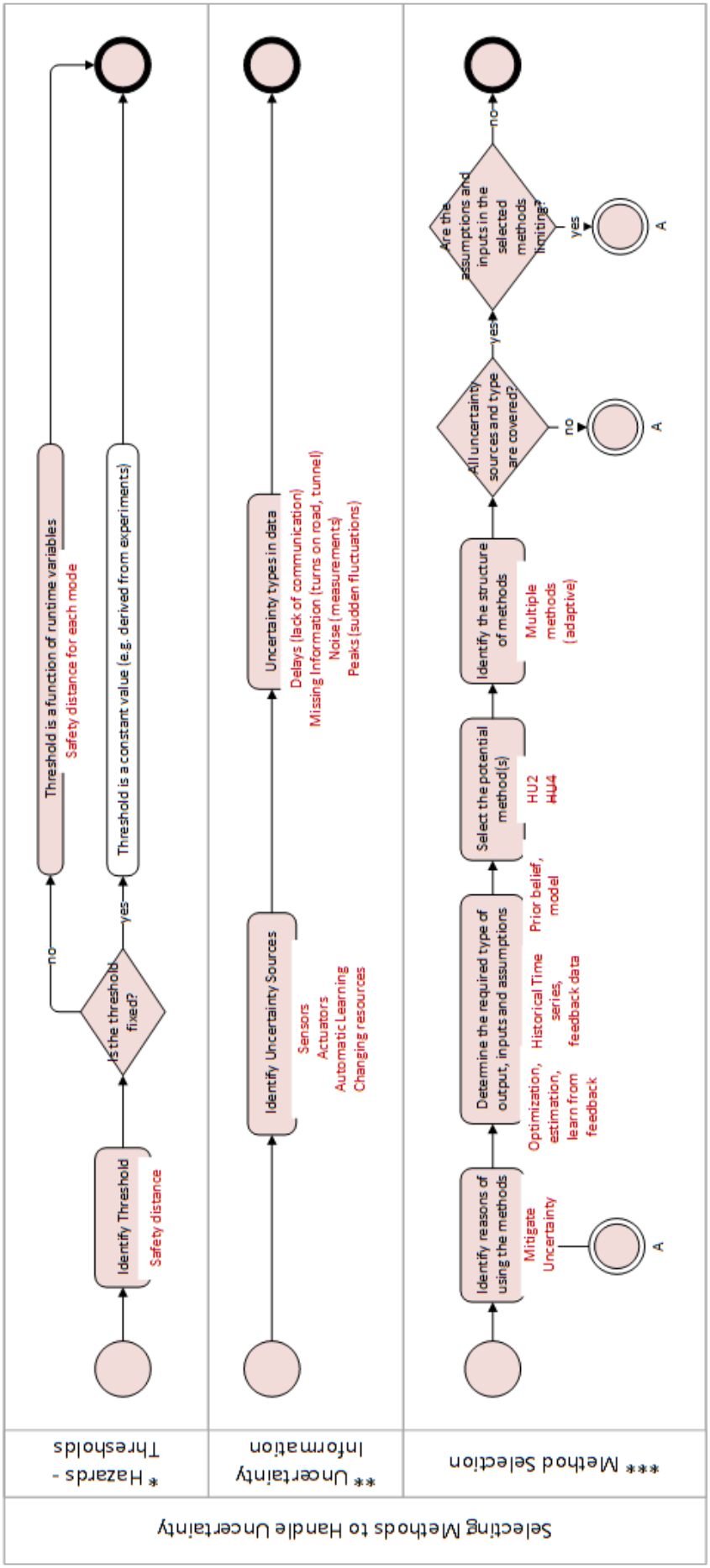


Figure 30: *Selecting platoon example details about hazards, uncertainty information, and methods.*

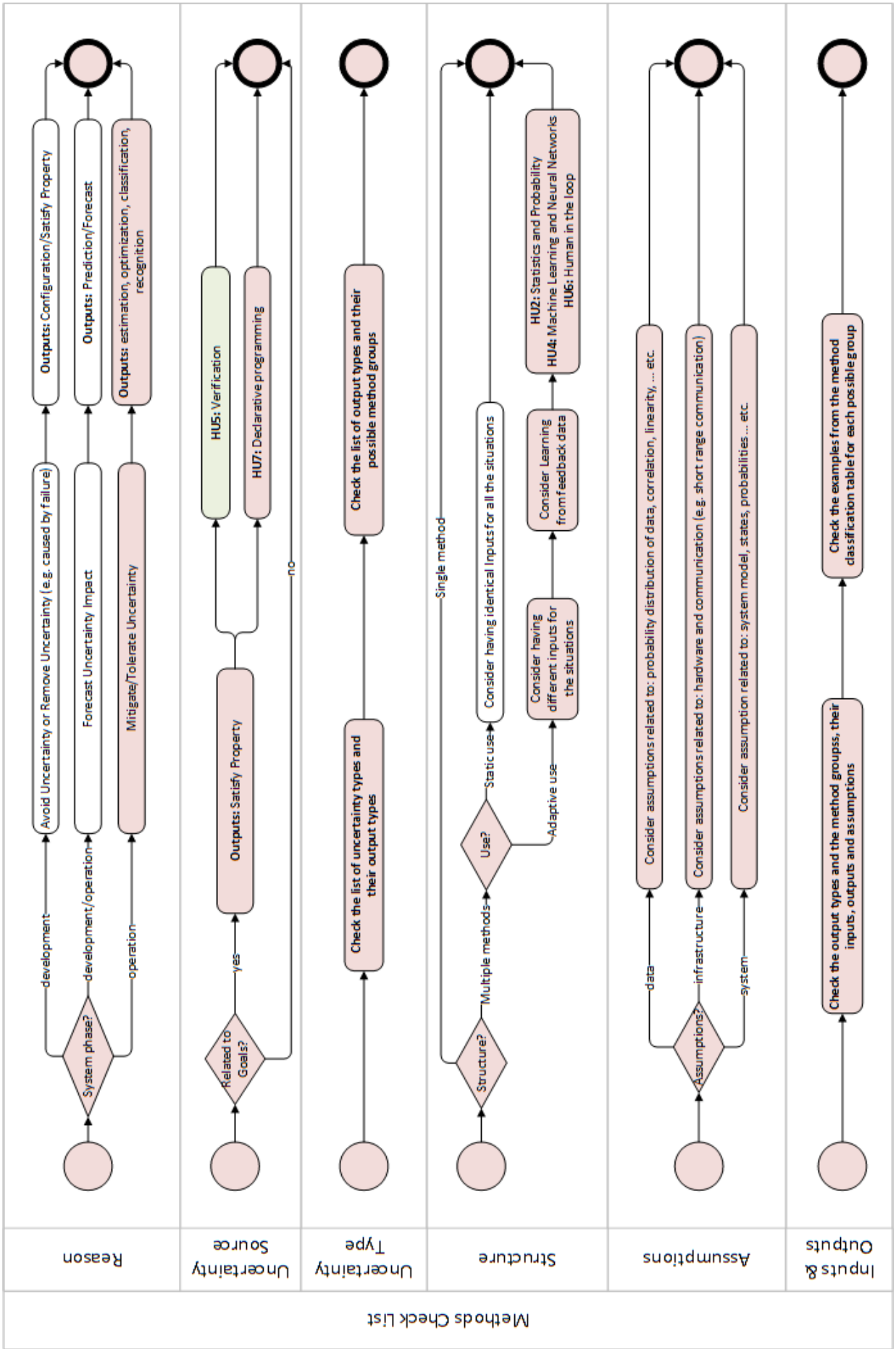


Figure 31: *The check list of method for platoon example*

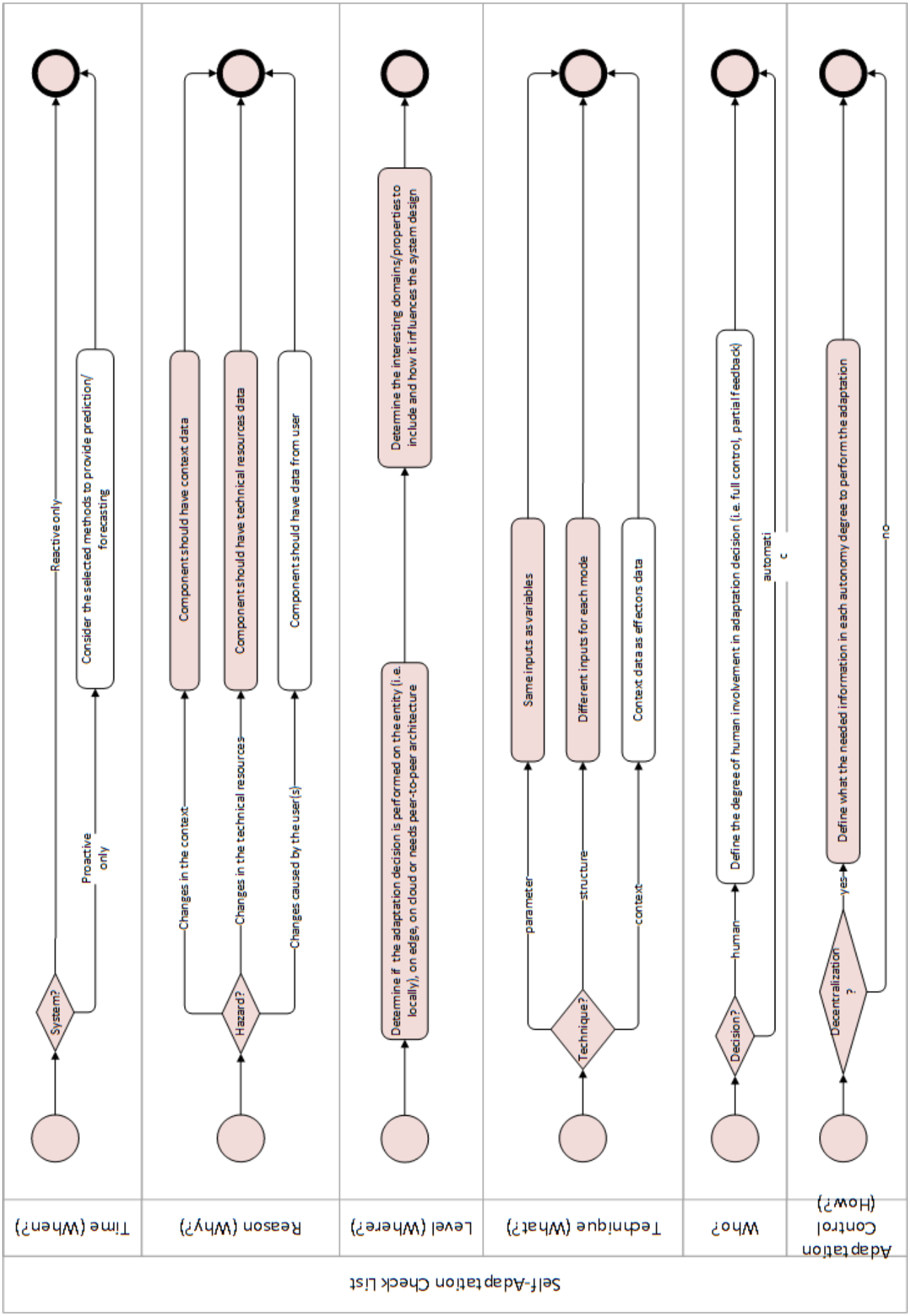


Figure 32: *The check list of self-adaptation for platoon example*