Grant Agreement Number: H2020-ICT-2014-1. 645463

Project Title: Testing Cyber-Physical Systems under Uncertainty: Systematic, Extensible, and Configurable Model-based and Search-based Testing Methodologies

Revision of deliverable report D1.2: Updated Report on U-Taxonomy

Written by: Simula Research Laboratory: Man Zhang, Bran Selic, Shaukat Ali, Tao Yue, Dipesh Pradhan (All the sections excluding Section 4 and Section 5)
Technical University Wien: Stefan Nastic, Hong-Linh Truong (Section 4)
Fraunhofer FOKUS: Martin Schneider, Max Bureck, Christian Hein (Section 5)

Project Acronym	U-TEST	**Grant Agreement Number**		H2020-ICT-2014-1. 645463	
Document Version	V2.0	**Date**	2015-06-30	**Deliverable No.**	1.2
Contact Person	Shaukat Ali	**Organisation**		Simula Research Laboratory	
Phone	+47 678282 00	**E-Mail**		shaukat@simula.com	

Executive Summary

Uncertainty is inherent in Cyber-Physical Systems (CPS). Dealing with uncertainty in CPS in a cost-effective manner is imperative for their reliable operations. Since designing, developing, and testing modern and highly sophisticated CPS is an expanding field, a step towards supporting handling uncertainty is to identify, define, and classify uncertainties at various levels of CPS. This will help develop a systematic and comprehensive understanding of uncertainty. To that end, we propose a taxonomy of uncertainty specifically designed for CPS. Since the study of uncertainty in CPS development and testing is still irrelatively unexplored, this taxonomy was derived in a large part by reviewing existing work on uncertainty in other fields, including philosophy, physics, statistics, and healthcare. The taxonomy is mapped to the three logical levels of CPS: Application, Infrastructure, and Integration.

Table of Contents

1 Introduction

Cyber-Physical Systems (CPS) are present in a wide range of safety/mission critical areas, such as in manufacturing, logistics, aerospace, and aeronautics [1-3], to mention but a few. For example, the current investment in existing CPS is estimated to be greater than 32 trillion dollars and is anticipated to exceed 82 trillion dollars by 2025 as reported in [4]. Given the pervasiveness of CPS and their criticality to the daily functioning of our society, it is vital for such systems to be able to operate in a safe and reliable manner.

However, since they generally function in the context of an inherently complex and unpredictable physical environment, a major difficulty with these systems is that they must be designed and operated in the presence of uncertainty. By *uncertainty* we mean here the lack of certainty (i.e., knowledge) about the timing and nature of inputs, the state of a system, a future outcome, as well as other relevant factors.

Even in the presence of uncertainty, reliability and safety of CPS cannot be given up. This means that an acceptable level of reliability and safety of CPSs must be maintained even when facing uncertainty. In this direction, as part of our project, we are aiming to develop, Model-Based Testing (MBT) techniques for CPS under uncertainty, which is a promising testing approach that focuses extensively on computer-driven models [5, 6]. MBT can provide rigorous, methodical, and automated testing, reducing the number of residual faults in a CPS and thereby improving overall CPS quality. However, to the best of our knowledge, there has not yet been a comprehensive study of MBT in the presence of uncertainty in the context of CPS.

As a first crucial step in such an investigation, we feel that it is necessary to understand the phenomenon of uncertainty and all its relevant manifestations. This means to systematically identify, classify and specify uncertainties that might arise at any of the three levels of CPS: *Application*, *Infrastructure*, and *Integration*. Based on studying and analyzing existing uncertainty taxonomies developed in other fields, including philosophy, physics, statistics and healthcare [7-10], we have defined an uncertainty taxonomy for CPS, which we call the *U-Taxonomy*. The objective is multifaceted: 1) to provide a unified and comprehensive description of uncertainties to both researchers and practitioners, 2) to classify uncertainties with the aim of identifying common representational patterns when modeling uncertain behaviors, 3) to provide a reference model for systematically collecting uncertainty requirements, 4) to serve as a methodological baseline for modeling uncertain behaviors in CPS, and, last but not least, 5) to provide a basis for standardization of MBT in the presence of uncertainty, leading to its broader application in practice.

U-Taxonomy is specified as a conceptual model comprising a set of UML class diagrams, which represent various uncertainty concepts, their attributes and relationships. These are complemented with a set of formal OCL constraints, which are enforced and validated automatically when the *U-Taxonomy* is instantiated. Each concept in the taxonomy was carefully selected and defined based on a thorough review of a broad spectrum of available literature on the topic (e.g., philosophy and physics) as well as our experience with working with CPSs in various domains. We illustrate the taxonomy using a running example of a videoconferencing system (VCS) with which we had prior experience. This case study has already been published in our earlier work [11]. Note that the examples in this document are specifically created to explain concepts and do not represent the real requirements of the system presented in [11].

We present details on the comprehensive validation of the *U-Taxonomy* in another deliverable, which is unfortunately not publicly available. In this document, we exclusively focus on presenting the final version of the taxonomy after validation. The rest of the document is organized as follows: Section 2 presents basic definitions, running example, and the structure of the *U-Taxonomy*. Section 3 introduces the Integration level taxonomy and how it links the Infrastructure level (Section 4) and Application level (Section 5) taxonomies. Section 6 provides related work, and the deliverable is concluded in Section 7.

2 Definitions, Running Example, and Introduction to U-Taxonomy

2.1 Definitions

This section presents necessary definitions.

Cyber-Physical Systems (CPS): In the context of our project, we defined CPS as follows: A set of heterogeneous physical units (e.g., sensors, control modules) communicating via heterogeneous networks (using networking equipment) and interacting with applications deployed on cloud infrastructures and/or humans to achieve a common goal. Conceptually a CPS is shown in Figure 1.

Levels of Uncertainties: Uncertainties can occur at three levels as shown in Figure 1.

- *Application Level:* At the application level, uncertainties are due to the events/data to/from a CPS from its application, e.g., a sensor sensing temperature.

- *Infrastructure Level:* Uncertainties that are occurring in the infrastructure of a CPS, e.g., in physical units, their interactions with communication networks, or in the cloud infrastructure.

- *Integration Level:* At this level, we classify uncertainties that happen due to the interactions among application and infrastructure level uncertainties or because of interactions among application and infrastructure level components.

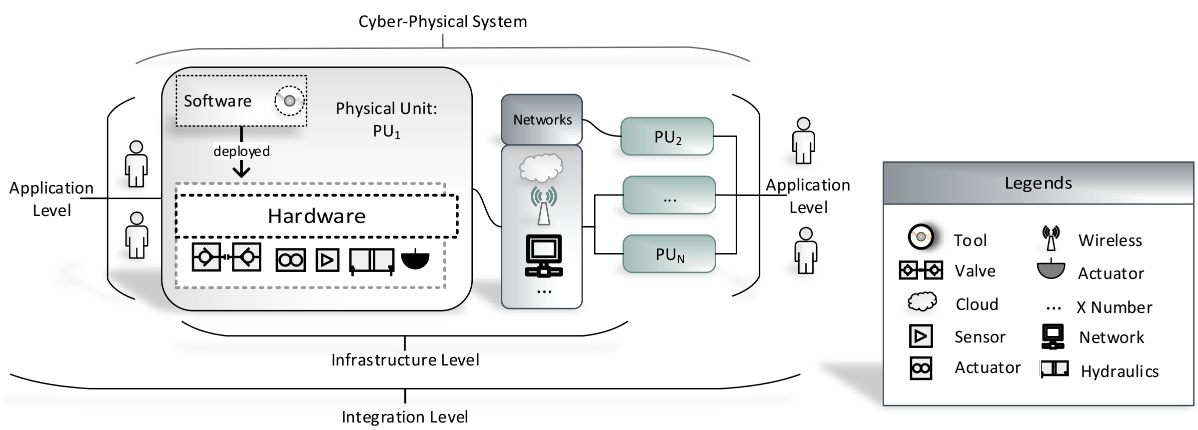

Figure 1. Cyber-Physical System

2.2 Running Example

The running example is a Videoconferencing System (VCS) [11] that was published in our earlier work, whose essential functionality is to make videoconference with a set of other systems that can be dedicated hardware-based VCSs, software-based VCSs for PCs, and cloud-based VCS solutions. Being an example of CPSs, it can experience uncertainties due to a variety of networks, cloud-based infrastructures, and a variety of other systems being in a videoconference. Note that examples are created solely to explain concepts and do not represent the real requirements of the system presented in [11].

2.3 Introduction to U-Taxonomy

The *U-Taxonomy* is an attempt to capture the core concepts associated with the notion of uncertainty in the context of CPSs for the needs of the U-Test project.

2.3.1 Representation

The *U-Taxonomy* is specified as a reference model and is represented by a hierarchy of related MOF models. The top-level structure of the *U-Taxonomy* model is shown in Figure 2. Notice that in the context of this deliverable the conceptual models of *U-Taxonomy* are represented as UML class diagrams; however, the actual implementation of *U-Taxonomy* will take place in Deliverable 4.1 and will be implemented in MOF according to the grant agreement.

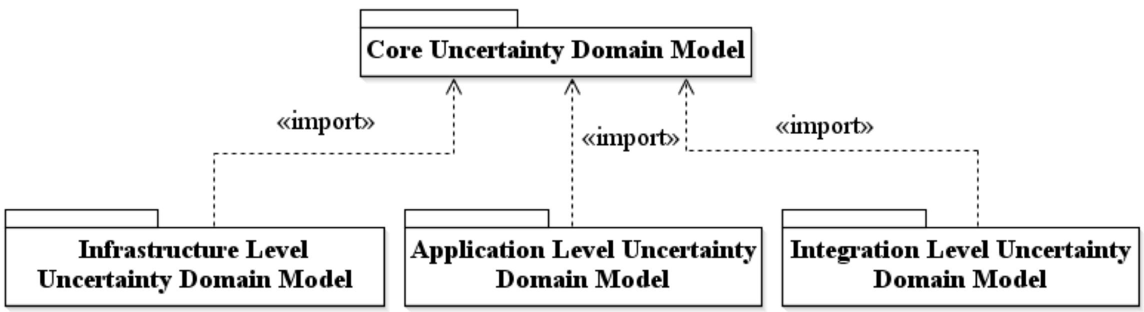

Figure 2. The top-level model of U-Taxonomy

The Core Uncertainty Domain Model captures the common abstractions that are further refined in three level-specific models, one for each level defined in the U-Test CPS conceptual framework (i.e., Infrastructure, Application, and Integration).

2.3.2 Specification Format

In this document the individual domain concepts of the reference model are described according to the following template:

- *Definition*: A succinct yet precise definition of the meaning of the concept.
- *Features*: A description of the properties (attributes) and associations belonging to the concept.
- *Semantics:* An informal description of the meaning of the concept.
- *Constraints:* Any constraints that apply to the concept; these are specified either using OCL or informal text (in cases where the OCL may be difficult to specify or interpret).
- *General examples* (optional)*:* One or more examples that illustrate the concept.
- *CPS-specific example* (optional): An example taken from the CPS domain.

For ease of reference, the concepts are listed in alphabetical order. The rest of the document is organized as follows: Section 3 presents the Integration level taxonomy, Section 4 presents Infrastructure level taxonomy, and Section 5 presents the Application level taxonomy.

3 The Core Uncertainty Domain Model (Integration Level Uncertainty Domain Model)

This top-level model captures the core abstractions of the *U-Taxonomy*. For convenience, it is partitioned into two areas: the *Core Belief Model* and the *Core Uncertainty Model*. The *Core Belief Model* deals with the concepts of belief agents and beliefs and their relationships to objective reality, which, naturally, serves as the reference against which to judge beliefs. The *Core Uncertainty Model* deals with the concepts of uncertainty and the various ways in which that concept is refined. The two models are coupled via the shared concept of *Uncertainty*.

3.1 The Core Belief Model

This section presents the core model of belief (shown in Figure 3) and is, consequently, referred to as the *Core Belief Conceptual Model*. It is inspired by the concepts defined in [12]. The *Core Belief Model* is kept as generic as possible. However, it is extensible and customizable to suit the specific requirements of the U-Test project. This means taking a viewpoint that focuses on generation of test cases and model evolution resulting from information gleaned as a result of exploring the ramifications of an uncertainty-focused system model (via the use of search-based techniques).

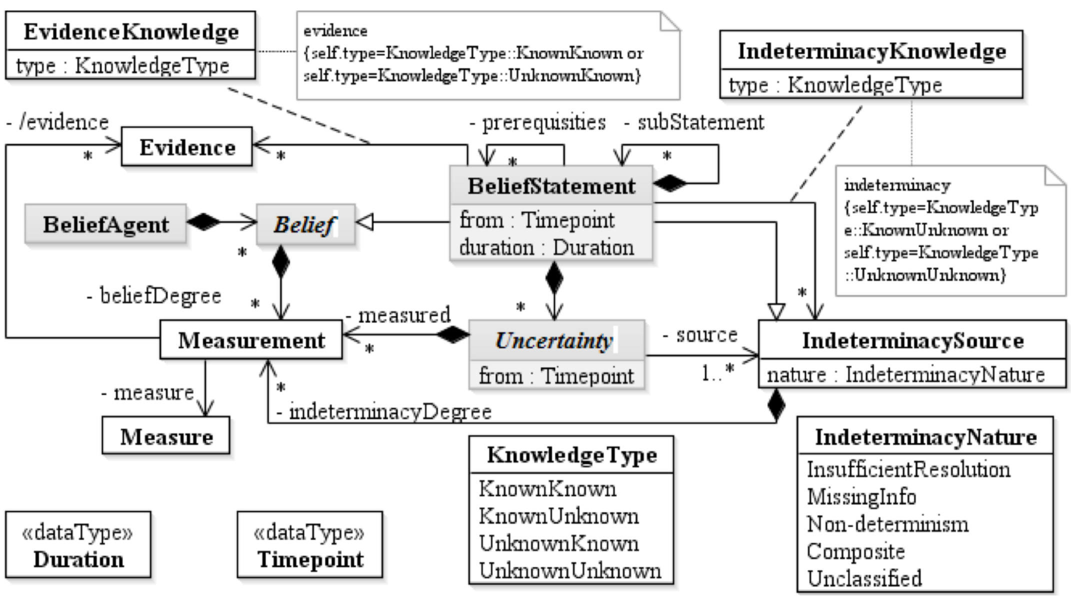

Figure 3. The Core Belief Model

The basic premise behind the model shown in Figure 3 is that uncertainty is a *subjective* phenomenon. That is, it is indelibly bound to the worldview held by a *belief agent*. A worldview of a belief agent can be represented as a set of *belief statements* about factual state of some phenomena or notions is unavailable. These statements may be explicit or implicit (e.g., existing in the mind(s) of the belief agent). Clearly, different belief agents may have very different views about the uncertainties associated with a particular belief statement – which is why each belief statement is intimately coupled to a particular belief agent.

Furthermore, these statements may be either valid or invalid, depending on whether or not they accurately represent objective reality[1].

At this point, it is important to note that a belief agent not necessarily needs to represent an individual; it might represent a community of individuals or even some technological system that is involved in decision-making, such as a specialized computer system[2].

The difference between *subjective* and *objective* concepts is important here: Subjective concepts are those that are based on potentially biased opinions of a belief agent. On the other hand, the form and value of objective concepts are determined by objective reality (*IndeterminacySource* and *IndeterminacyNature*) and are, therefore, independent of any opinions by held by belief agents. In the diagram above, the uncoloured elements represent objective concepts, whereas the grey elements represent subjective concepts.

Uncertainty in this model represents a state of affairs whereby a belief agent does not have full confidence in a held belief due to any of a number of factors: lack of information, the Heisenberg uncertainty principle in physics, conscious ignorance of *BeliefAgent*, some inherent variability in the domain, or other possible reasons. The *degree of uncertainty* of a belief statement can change over time as more information becomes available[3].

It is important to note that the relationship between full confidence in a belief and objective reality (i.e., *the truth*) is not necessarily straightforward: a belief agent may have full confidence in a belief that does not actually correspond to the truth[4]. Furthermore, as the state of the subject area may change for dynamic systems, this relationship can also change over time. Thus, a belief that may have been perfectly *valid* at one point in time may not remain so forever. Hence, even beliefs that are deemed “proven” by a belief agent (e.g., based on some other available *Evidence*), should not necessarily be assumed to be valid; it may depend on the point in time when the validity assessment is made.

3.1.1 Belief

Definition	A belief is an *implicit* subjective explanation or description of some phenomena or notions[5] that is held by a BeliefAgent.
Features	• agent – The BeliefAgent who holds the belief represented by the BeliefStatement. • beliefDegree – This Measurement is used for representing confidence degree from BeliefAgent held this Belief.
Semantics	This is an *abstract concept* whose only concrete manifestation is in the form of a belief statement.
Constraints	• Each Measurement owned by one Belief is different type of Measure.

[1] Strictly speaking, such a strictly binary categorization may not be always realistic, since beliefs could be characterized by degrees of truthfulness. However, in this model, we choose to ignore such subtleties. A belief statement is deemed to be valid if it is a sufficient approximation of the truth for the purpose on hand.
[2] In this case, the beliefs would be reflected in the rules that are programmed into the system.
[3] In fact, the degree of uncertainty may even increase as a result of additional pertinent information becoming available that exposes previously trusted beliefs to be less certain.
[4] For example, many people in the past were absolutely certain that the Earth was flat and that it was the centre of the Universe.
[5] The term “phenomena” here is intended to cover aspects of objective reality, whereas “notion” covers abstract concepts, such those encountered in mathematics or philosophy.

	`context Belief inv: self.beliefDegree->size() > 1 implies (self.beliefDegree->one(m:Measurement|m.measure.oclIsKindOf(UTaxonomy::MeasureModel::Probability)) or self.beliefDegree->one(m:Measurement|m.measure.oclIsKindOf(UTaxonomy::MeasureModel::Ambiguity)) or self.beliefDegree->one(m:Measurement|m.measure.oclIsKindOf(UTaxonomy::MeasureModel::Vagueness)))`

3.1.2 BeliefAgent

Definition	BeliefAgent represents an individual, a community of individuals sharing the same set of beliefs, or a technology, such as a software system, with built-in beliefs.
Features	• beliefs – The set of Belief that represent the full set of beliefs held explicitly by the BeliefAgent.
Semantics	A belief agent is a physical entity[6] that holds (i.e., owns) one or more beliefs about phenomena or notions associated with one or more subject areas derived from Indeterminacy. This could be a human individual or group, an institution, a living organism, or even a machine such as a computer. Crucially, a belief agent is capable of actions based on its beliefs.
Constraints	None.
General Example	A philosopher; a software program that performs weather prediction.
CPS Example	A VCS system tester.

3.1.3 BeliefStatement

Definition	A BeliefStatement is an *explicit* specification of some Belief held by a BeliefAgent about a possible phenomenon or notions belonging to a given subject area.
Features	(inherited from IndeterminacySource and Belief parent) • substatements – The set of finer-grained BeliefStatements that are components of a composite BeliefStatement (e.g., "there are three balls in the box and one of them is red" clearly decomposes into two distinct finer-grained BeliefStatements). • indeterminacySource – The set of IndeterminacySource that this BeliefStatement involves (note that since BeliefAgents have imperfect knowledge of Indeterminacy, they may not be fully aware of which Indeterminacy are actually covered by a particular BeliefStatement). • prerequisites – The set of BeliefStatement on which this BeliefStatement depends. • from – The Timepoint when BeliefStatement is initialized. • duration – The Duration when BeliefStatement is active. • uncertainty – The set of expressions of uncertainty that qualify and/or quantify the degree to which the BeliefAgent lacks confidence in this BeliefStatement; this attribute provides the core link between the Belief portion and the Uncertainty portion of the core uncertainty model (see

[6] We exclude here from this definition "virtual" belief agents, such as those that might occur in virtual reality systems and computer games.

	Section 3.2).
Semantics	The concrete form of this statement can vary, and may represent informal pronouncements made by individuals or groups, documented textual specifications expressed in either natural or formal languages, formal or informal diagrams, etc. Since it represents a belief, which is a subjective concept, a BeliefStatement may not necessarily correspond to objective reality. This means that it could be completely false, or only partially true, or completely true. However, due to the complex nature of objective reality, it may not always be possible to determine whether or not a BeliefStatement is valid. Furthermore, the validity of a statement may only be meaningfully defined within a given context or purpose. Thus, the statement that "the Earth can be represented as a perfect sphere" may be perfectly valid for some purposes but invalid or only partly valid for others. For our needs, we are less interested in the validity of a BeliefStatement than we are in the level of Uncertainty that a belief agent associates with it.
Constraints	None.
General Example	For purposes of general astronomy, the Earth can always be considered as a perfect sphere.
CPS Example	The worst-case packet loss rate in a VCS during a videoconference will never exceed 3.2%.

3.1.4 Evidence

Definition	Evidence is either the observation of or record of a real-world event occurrence or, alternatively, the conclusion of some formalized chain of logical inference, which provides information that may contribute to determining the validity (i.e., truthfulness) of a BeliefStatement.
Features	None.
Semantics	Evidence is fundamentally an objective phenomenon, representing *something that actually happened*. This means that we do exclude here the possibility of counterfeit or invented evidence. Nevertheless, although Evidence represents objective reality, it need not be conclusive in the sense that it removes all doubt (uncertainty) about a BeliefStatement. On the other hand, any valid BeliefStatement must have at least some Evidence to support it.
Constraints	None.
General Example	Rainfall was recorded at 1 PM today, providing evidence to the belief that it rained today.
CPS Example	The regression tests demonstrated that the VCS system meets its functional requirements.

3.1.5 EvidenceKnowledge

Definition	EvidenceKnowledge expresses an objective relationship between a BeliefStatement and relevant Evidence.
Features	• type – this value represent the knowledge relationship between BeliefStatement held by BeliefAgent and Evidence.
Semantics	Evidence identifies whether the corresponding BeliefAgent is aware of the appropriate Evidence. Thus, an agent may be either aware that it knows something (KnownKnown), or it may be completely unaware of Evidence (UnknownKnown). This is formally expressed by the two constraints attached to

	EvidenceKnowledge

Constraints	• The type attribute attached on EvidenceKnowledge should be KnownKnown or UnknownKnown
	`context EvidenceKnowledge`
	`inv: self.type = KnowledgeType::KnownKnown or self.type = KnowledgeType::UnknownKnown`

General Example	The observation of yesterday's rainfall from BeliefAgent is Evidence, and the relationship between it and "It will rain more than 3 times in this year." This BeliefAgent made is KnownKnown EvidenceKnowledge.
CPS Example	The results of testing dial function known by Software Testing Engineer is Evidence, and the relationship between it and "It will dial successfully 80% of time" this Software Testing Engineer made is KnownKnown EvidenceKnowledge.

3.1.6 Indeterminacy

Definition	Indeterminacy represents a situation whereby the full knowledge necessary to determine the required factual state of some phenomena or notions is unavailable[7].
Features	None.
Semantics	This is an *abstract concept* whose only concrete manifestation is in the form of an IndeterminacySource. This may be due to either subjective reasons (e.g., agent ignorance) or objective reasons (e.g., the Heisenberg uncertainty principle in physics). The kinds of Indeterminacy are captured as IndeterminacyNature (see Section 3.1.8). It may be useful to identify explicitly, what are the factors that lead to uncertainty on behalf of the BeliefAgent. These could be multifarious and are referred to as IndeterminacySources in the metamodel.
Constraints	None.
General Example	Weather (in the future)
CPS Example	Packet loss (during videoconference)

3.1.7 IndeterminacyKnowledge

Definition	IndeterminacyKnowledge expresses an objective relationship between an IndeterminacySource and the awareness that the BeliefAgent has of that source.
Enumeration literals	• type – this value represents the knowledge relationship between BeliefStatement held by BeliefAgent and IndeterminacySource.
Semantics	None.

Constraints	• The type attribute attached on IndeterminacyKnowledge should be KnownUnknown or UnknownUnknown
	`context IndeterminacyKnowledge`
	`inv: self.type = KnowledgeType::KnownUnknown or self.type = KnowledgeType::UnknownUnknown`

[7] Care should be taken to distinguish between indeterminacy and non-determinism. The latter is only one possible source of indeterminacy.

General Example	Unpredicted rainfall in the rest of this year known by this BeliefAgent is IndeterminacySource, and the relationship between it and “It will rain more than 3 times in this year.” this BeliefAgent made is KnownUnknown IndeterminacyKnowledge.
CPS Example	The randomness of successful dial at runtime by Software Testing Engineer is Indeterminacy, and the relationship between it and “It will dial successfully 80% of time” this Software Testing Engineer made is KnownUnknown IndeterminacyKnowledge.

3.1.8 IndeterminacyNature (Enumeration)

Definition	IndeterminacyNature represents the kind of Indeterminacy.
Enumeration literals	• InsufficientResolution – The information available about the phenomenon in question is not sufficiently precise. • MissingInfo – The full set of information about the phenomenon in question is unavailable at the time when the statement is made. • Non-determinism – The phenomenon in question is either practically or inherently non-determinism. • Composite – This represents a combination of more than one kinds of indeterminacy. • Unclassified – Indeterminate indeterminacy.
Semantics	See 3.1.6 Indeterminacy.
Constraints	None.
General Example	Heisenberg uncertainty (Non-determinism).
CPS Example	Congested status of Network (Non-determinism).

3.1.9 IndeterminacySource

Definition	IndeterminacySource represents a situation whereby the information required to ascertain the validity of a BeliefStatement is indeterminate in some way, resulting in uncertainty being associated with that statement.
Features	• indeterminacyDegree – This set of Measurement represents the quantification (or qualification) of this IndeterminacySource. • nature – The set of IndeterminacyNature represents the kind of indeterminacy reason.
Semantics	One possible source of indeterminacy could be another BeliefStatement (which is why the latter is shown as a specialization of IndeterminacySource in Figure 3). A given indeterminacy source could in some cases be decomposed into more basic sources.

Constraints	• Each Measurement owned by one IndeterminacySource is different type of Measure.
	`context IndeterminacySource`
	`inv: self.indeterminacyDegree->size() > 1 implies (self. indeterminacyDegree ->one(m:Measurement|m.measure.oclIsKindOf (UTaxonomy::MeasureModel::Probability)) or self. indeterminacyDegree ->one(m:Measurement|m.measure.oclIsKindOf(UTaxonomy::MeasureModel::Ambiguity)) or self. indeterminacyDegree -> one(m: Measurement|m.measure.oclIsKindOf(UTaxonomy::MeasureModel::Vagueness)))`

General Example	Unreliable Human behaviour in the future (Non-determinism).

CPS Example	Unpredicted status of Network at runtime during videoconference (Non-determinism).

3.1.10 KnowledgeType (Enumeration)

Definition	KnowledgeType is captured in the model as an enumeration with four values:
Enumeration literals	• KnownKnown – Indicates that an associated BeliefAgent is consciously aware of some relevant aspect. • KnownUnknown (Conscious Ignorance) – Indicates that an associated BeliefAgent understands that it is ignorant of some aspect. • UnknownKnown (Tacit Knowledge) – Indicates that an associated BeliefAgent is not explicitly aware of some relevant aspect that it, nevertheless, may be able to exploit in some way • UnknownUnknown (Meta Ignorance) – Indicates that an associated BeliefAgent is unaware of some relevant aspect.
Semantics	At a given point in time, a BeliefAgent always makes a statement based on a KnownKnown Evidence and a KnownUnknown IndeterminacySource. Splitting EvidenceKnowledge and IndeterminacyKnowledge provides the flexibility to enable transitions among different knowledge types (e.g., from UnknownKnown to KnownKnown), based on the evolution of EvidenceKnowledge and IndeterminacyKnowledge related to the associated BeliefAgent.
Constraints	None.
General Example	-
CPS Example	-

3.1.11 Uncertainty in Belief Model

Definition	Uncertainty is a state of a BeliefAgent whereby the agent does not have full confidence in the validity of a belief statement
Features	• from – The Timepoint when BeliefStatement is initialized. • measured – This value is used for representing confidence degree of uncertainty by the agent making the BeliefStatement. • uncertainty – The set of Uncertainty specifying this Uncertainty is related to. • source – This set of IndeterminacySource derived from the involves association and generalization of BeliefStatement.
Semantics	"Full confidence" here means that the agent does not have any doubts about the validity of a statement. It is important to distinguish here between certainty and validity. That is, an agent could have full confidence in a BeliefStatement that is actually false; i.e., a statement that does not match (objective) truth. In general, the source of uncertainty associated with a BeliefStatement is that, for some reason, the agent does not have full knowledge of all relevant facts pertaining to the phenomena or notions that are the subject of the statement (see Indeterminacy).

Constraints	• Each Uncertainty has at least one IndeterminacySource.
	`context Uncertainty`
	`inv: self.source->size() >= 1`
	• The source of Uncertainty is sub or equal set of those BeliefStatement

	involves.
	`context BeliefStatement`
	`inv: self.uncertainty->forAll(u:Uncertainty| self.indeterminacySource->includesAll(u.source))`
	• Each Measurement owned by one Uncertainty is different type of Measure.
	`context Uncertainty`
	`inv: self.measured->size() > 1 implies (self.measured->one(m:Measurement|m.measure.oclIsKindOf(UTaxonomy::MeasureModel::Probability)) or self.measured->one(m:Measurement|m.measure.oclIsKindOf(UTaxonomy::MeasureModel::Ambiguity)) or self.measured->one(m:Measurement|m.measure.oclIsKindOf(UTaxonomy::MeasureModel::Vagueness)))`

General Example	I believe that I will rain tomorrow with 78% probability.
CPS Example	I believe that the rate of packet loss during videoconference is less than 3% with 90% probability.

Further treatment of this concept is defined in Section 3.2.9.

3.1.12 Measure

Definition	Measure represents the measured way of Uncertainty.
Features	None.
Semantics	Measure is *objective concept*, and specifies the existing way/theory to measure uncertainty.
Constraints	None
General Example	Probability
CPS Example	-

Further treatment of this concept is defined in Section 3.2.10.

3.1.13 Measurement

Definition	Measurement when associated with a given IndeterminacySource represents the optional quantification (or qualification) that specifies the degree of indeterminacy of the IndeterminacySource.
Features	• measure – This value represents the related way of measuring uncertainty.
Semantics	It may be possible to specify a Measurement that quantifies in some way (e.g., as a probability or a percentage) the degree of uncertainty by the agent making the belief statement. Note, however, that this is a subjective measure defined by the BeliefAgent.
Constraints	None.
General Example	"The probability that it will rain tomorrow in Oslo is 30%". In this example, uncertainty is measured as a probability value, i.e., 30% by probability measuring method.
CPS Example	"The probability that packet loss during a videoconference is less than 3.2% is 85%".

Further treatment of this concept is defined in Section 3.2.10.2.

3.2 The Core Uncertainty Model

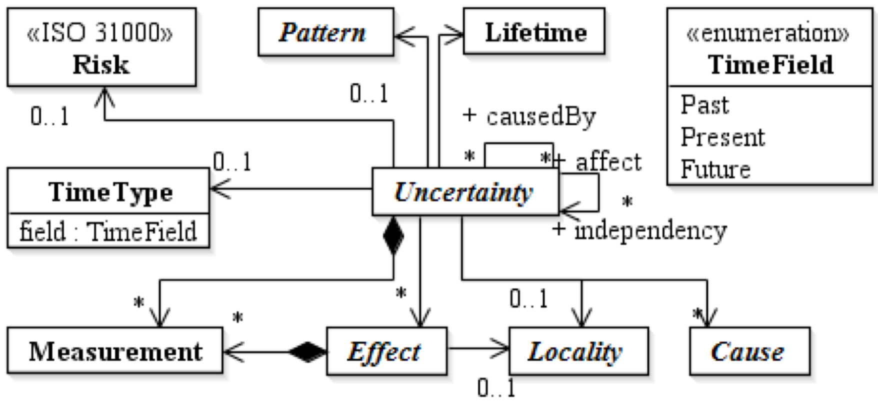

Figure 4. The Core Uncertainty Model

This model (shown in Figure 4) inspired from the concepts defined in [12-16] is an adjunct to the Core Belief model described in the previous section. It expands on the concept of Uncertainty from different viewpoints and introduces related abstractions. These additional concepts in the core uncertainty model link application and infrastructure level taxonomies to the integration level taxonomy. Notice that the *Uncertainty* concept has a self-association. This self-association facilitates: 1) relating different application level uncertainties to each other, 2) relating different infrastructure level uncertainties to each other, 3) relating application level and infrastructure level uncertainties to each other.

3.2.1 Cause

Definition	Anything from which an Uncertainty occurs in the BeliefStatement.
Features	
Semantics	The cause for an Uncertainty can be: 1) another known Uncertainty, 2) something known and is not Uncertainty, 3) anything unknown. If a Cause is Uncertainty, then it may be measured using Measurement.

Constraints	• Any Cause causes at least one Uncertainty.
	`context Cause`
	`inv: Uncertainty.allInstances()->one(u: Uncertainty| u.cause->includes (self))`

General Example	"Holding the outdoor activity depends on whether it will rain tomorrow". In this BeliefStatement, rain is the Cause, whereas there is Uncertainty in the Occurrence of the outdoor activity.
CPS Example	The VCS endpoint cannot make a call due to improper human behaviour, where he/she didn't enter the complete number of the VCS to dial. Human behaviour is the Cause of Uncertainty, whereas entering incomplete number is Non-determinism of IndeterminacyNature (see Section 3.1.8).

3.2.2 Effect

Definition	Effect represents the result of Uncertainty in the BeliefStatement.
Features	• locality – This value is used to represent that the Locality (See Section 3.2.4) of the Effect. • measurements – This value is used for representing what kind of measurement may be used to measure this Effect.

Semantics	An uncertainty may result into: 1) another known Uncertainty, 2) something known and is not Uncertainty, 3) anything unknown.

Constraints	• Any Effect is produced by at least one Uncertainty.
	`context Effect`
	`inv: Uncertainty.allInstances()->one(u: Uncertainty| u.effect->includes (self))`
	• Each Measurement owned by one Effect is different type of Measure.
	`context Effect`
	`inv: self.measurement->size() > 1 implies (self.measurement->one(m: Measurement|m.measurement.oclIsKindOf(UTaxonomy::MeasureModel::Probability)) or self.measurement->one(m:Measurement|m.measure.oclIsKindOf (UTaxonomy ::MeasureModel::Ambiguity)) or self.measurement->one(m: Measurement |m.measurement.oclIsKindOf(UTaxonomy::MeasureModel::Vagueness)))`

General Example	"The uncertainty in weather may affect the flight landing". In this BeliefStatement, the Effect of Uncertainty, i.e., weather conditions, is on the flight landing.
CPS Example	"The uncertainty in the percentage of packet loss affects the videoconference quality". In this BeliefStatement, the Effect of Uncertainty, i.e., percentage of packet loss, is on the quality of videoconference.

3.2.3 Lifetime

Definition	Lifetime represents the duration of time for which an Uncertainty remains active.
Features	None.
Semantics	The length of time for which Uncertainty exists. For example, an Uncertainty may appear temporarily for a short period of time and disappears itself. On the other hand, an Uncertainty could be persistent, i.e., it stays active until appropriate actions are taken to resolve the Uncertainty.

Constraints	• An Uncertainty must have exactly one lifetime.
	`context Uncertainty`
	`inv: self.lifetime->notEmpty()`

General Example	"The weather forecast of tomorrow in Oslo is uncertain from 4PM to 5PM". In this example, the uncertainty's Lifetime is tomorrow.
CPS Example	"The packet loss of approximately 1.5% occurs during videoconference". In this case, the Lifetime of packet loss is the period of videoconference.

3.2.4 Locality

Definition	A particular place or a position where Uncertainty occurs in the BeliefStatement.
Features	None.
Semantics	A location could be a geographical location or a position where Uncertainty occurs. The concept of location is different than the Uncertainty type GeographicalLocation (Section 3.2.9.3), where Uncertainty is due to the geographical location, however in this concept Uncertainty occurred at a location may not be due to the geographical location.
Constraints	None.
General Example	"It will rain tomorrow in Oslo". In this example, Locality of Uncertainty is the exact location in Oslo, where it will rain.
CPS Example	In our example of packet loss, the Locality of Uncertainty is in the infrastructure level.

3.2.5 Pattern

Figure 5 shows a conceptual model for the occurrence Pattern of Uncertainty inspired from the concepts reported in [15, 17, 18]. Notice that in this section, the patterns presented are by no means representation of a complete set of patterns that may exist for Uncertainty. These patterns are the commonly known patterns.

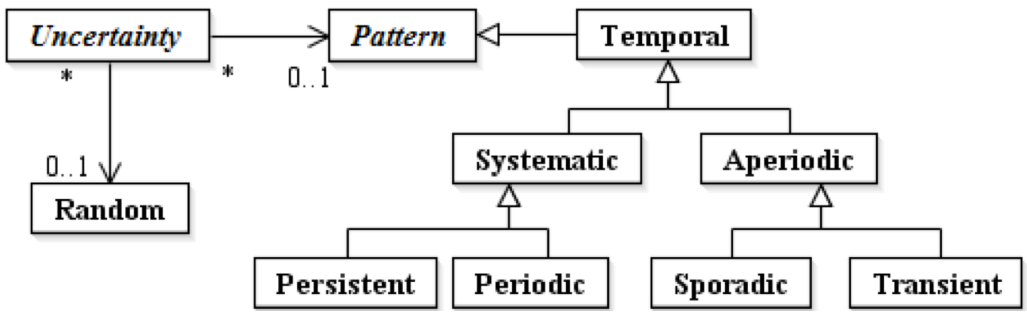

Figure 5. The Pattern of Uncertainty

Definition	Pattern represents an intelligible way in which an Uncertainty appears.
Features	None.
Semantics	An Uncertainty may occur without any Pattern, i.e., Random (Section 3.2.5.4), or may have a pattern in which it may occur, for example, occurring at equal intervals of time, i.e., Periodic (Section 3.2.5.2).
Constraints	None.
General Example	-
CPS Example	-

3.2.5.1 Aperiodic

Definition	An Uncertainty that occurs at irregular intervals of time.
Features	(inherited from Temporal parent)
Semantics	It is important to note that Aperiodic is inherited from Temporal; this means it has a notion of time in which the Uncertainty appears in an Aperiodic pattern.
Constraints	None.
General Example	-
CPS Example	-

3.2.5.2 Periodic

Definition	An Uncertainty that occurs in repeated periods or at regular intervals.
Features	(inherited from Systematic parent)
Semantics	Uncertainty repeating itself after an equal interval of time.
Constraints	None.
General Example	-
CPS Example	-

3.2.5.3 Persistent

Definition	A permanent Uncertainty, i.e., lasting forever.
Features	(inherited from Systematic parent)
Semantics	The definition of "forever" varies. For example, an uncertainty may exist

	permanently until appropriate actions are taken to deal with the uncertainty. On the other hand, an uncertainty may not be able to resolve and stays forever.
Constraints	None.
General Example	-
CPS Example	-

3.2.5.4 Random

Definition	An Uncertainty that occurs without definite method, purpose or conscious decision.
Features	None.
Semantics	An Uncertainty occurring without any specific pattern.
Constraints	None.
General Examples	-
CPS Example	-

3.2.5.5 Temporal

Definition	Uncertainty occurring in a temporal pattern.
Features	(inherited from Pattern parent)
Semantics	Temporal describes the notion of time with the occurrence of uncertainty
Constraints	None.
General Example	-
CPS Example	-

3.2.5.6 Systematic

Definition	Uncertainty occurring in a systematic pattern.
Features	(inherited from Temporal parent)
Semantics	Uncertainty occurring in some methodical pattern, i.e., a pattern that can be described in a mathematical way.

Constraints	• An Uncertainty occurring in a systematic pattern has at least one Measurement.
	`context Uncertainty`
	`inv: self.pattern.oclIsTypeOf(Systematic) and self.measured-> notEmpty()`

General Example	-
CPS Example	-

3.2.5.7 Sporadic

Definition	Uncertainty occurring in a sporadic pattern.
Features	(inherited from Aperiodic parent)
Semantics	Uncertainty occurring occasionally.
Constraints	None.
General Example	"It rains sporadically in Islamabad"
CPS Example	"The packet loss occurs sporadically during a videoconference".

3.2.5.8 Transient

Definition	Uncertainty occurring temporarily.
Features	(inherited from Aperiodic parent)
Semantics	Uncertainty that doesn't last long.
Constraints	None

General Example	-
CPS Example	-

3.2.6 Risk

Figure 6 shows a conceptual model for the Risk associated with Uncertainty inspired from the concepts reported in [19-22]. Notice that we present one way of measuring risk here and other ways could be used for the measurement of risk associated with Uncertainty.

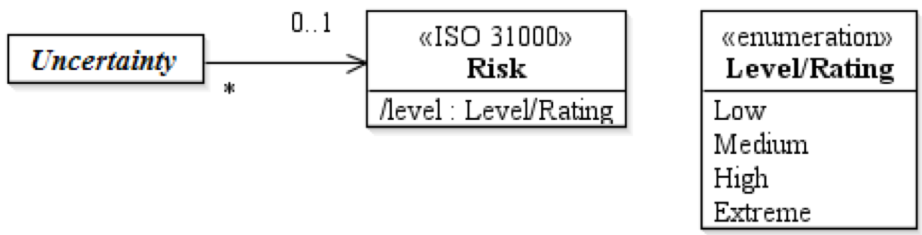

Figure 6. The Risk of Uncertainty

Definition	Risk measures the risk associated with Uncertainty.
Features	level – This value derived from Likelihood and Impact using risk matrix [19].
Semantics	An uncertainty may have an associated risk and high risk uncertainties deserve special attention.
Constraints	None.
General Example	-
CPS Example	-

3.2.6.1 Level/Rating

Definition	Risk level associated with an uncertainty.
Features	None
Semantics	Level/Rating is derived from Measurement owned by Uncertainty (Probability of the Occurrence of an Uncertainty) and Measurement owned by Effect (e.g., high impact), for example, using the risk matrix [19] or any other matrices.
Constraints	None.
General Example	-
CPS Example	-

3.2.7 TimeField

Definition	TimeField represents a point in time in past, present, or future about Uncertainty in the BeliefStatement.
Features	• field – This value is used for identifying a relative point in time.
Semantics	TimeField represents whether an Uncertainty occurred in the past, or is occurring in present, or will occur in the future. This concept is different than Lifetime, which captures a period of time for which an Uncertainty remains active.
Constraints	None.
General Example	"BBC estimated that it would rain approximately 8mm in Oslo tomorrow". In this example, Content of 8mm rain is uncertainty in the future. "It rained probably yesterday". In this example, Occurrence is uncertainty in the past.
CPS Example	"An estimated packet loss for videoconferences taking place from 10 AM to 11

	AM for tomorrow is 2.2%". The type of uncertainty is Content Uncertainty and is in future.

3.2.8 TimeType (Enumeration)

Definition	This enumeration is used to classify time, i.e., past, present, and future.
Enumeration literals	• Past – This Uncertainty occurred in the past. • Present – This Uncertainty is occurring at the present. • Future – This Uncertainty will occur in the future.
Semantics	There are no further semantics associated with TimeType.
Constraints	None.
General Example	-
CPS Example	-

3.2.9 Uncertainty

The types of uncertainty are present in Figure 7.

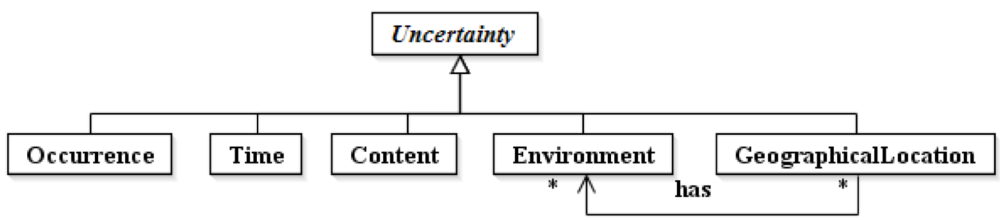

Figure 7. The Type of Uncertainty

Definition	Uncertainty represents a situation whereby a BeliefAgent lacks confidence in a BeliefStatement. Figure 7 shows a conceptual model for different types of Uncertainty inspired from the concepts reported in [12, 15, 16].
Features	• from – see Section 3.1.11. • measured – see Section 3.1.11. • lifetime – This value is used for representing the duration of this Uncertainty. • timeType – This value is used for time when this Uncertainty exists. • pattern – This value is used for describing whether this Uncertainty happens in a pattern or what kind of the pattern this Uncertainty occurs in. • risk – This value is used for whether this Uncertainty has a risk, and what kind of risk this Uncertainty causes. • locality – This value is used for representing what location this Uncertainty occurs. • causedBy – This value is used for representing the set of other Uncertainties induced by the presence of this Uncertainty. • affect – This value is used for representing the set of other Uncertainties which induces the presence of this Uncertainty. • effect – This value is used for describing what effect the Uncertainty may produce. • cause – This value is used for describing what cause the Uncertainty.

	• Independency – The set of Uncertainty represents the independency relationship with other Uncertainty. • beliefStatement – The BeliefStatement to which this Uncertainty applies.
Semantics	In principle, there could be multiple expressions of Uncertainty about a given BeliefStatement, although, in practice there is at most one. Multiple expressions might be used in cases where different ways of representing uncertainty are used concurrently. Note that uncertainty is a strictly subjective phenomenon and is indelibly associated with a BeliefStatement. It is the BeliefAgent that defines the level of uncertainty associated with such a statement.
Constraints	None.
General Example	"The probability that it will rain tomorrow during the afternoon is 20% (i.e., an uncertainty value of 20/100)".
CPS Example	"The likelihood that the VCS system will perform without any packet loss during a 30-minute videoconference is 50%".

3.2.9.1 Content

Definition	Content represents a situation whereby a BeliefAgent lacks confidence in content existing in a BeliefStatement.
Features	(inherited from Uncertainty parent)
Semantics	There are no further semantics associated with this concept.
Constraints	None.
General Example	"The rainfall capacity is at least 0.1mm/hr tomorrow. (i.e., an uncertainty value of more than 0.1mm/hr)".
CPS Example	"The packet loss during a videoconference is less than 3.2%, but it is possible that the packet loss is more than 3.2% due to unreliable network environment".

3.2.9.2 Environment

Definition	Environment represents a situation whereby a BeliefAgent lacks confidence in environment existing in a BeliefStatement.
Features	(inherited from Uncertainty parent)
Semantics	There are no further semantics associated with this concept.
Constraints	None.
General Example	"Depending on the environment conditions, there might be uncertainty in the weather forecast".
CPS Example	"The unexpected packet loss during a videoconference occurs under congested network".

3.2.9.3 GeographicalLocation

Definition	GeographicalLocation represents a situation whereby a BeliefAgent lacks confidence in geographical location existing in a BeliefStatement.
Features	(inherited from Uncertainty parent) • environment – The set of Environment represents the surroundings of this GeographicalLocation.
Semantics	There are no further semantics associated with this concept.
Constraints	None.
General Example	"It will rain in Oslo or Sweden tomorrow".
CPS Example	"The sound during a videoconference is continuous, but this is not always the

	case in Chinle and Fort Defiance." Notice that based on statistics these two cities have the worst internet connection in the USA.

3.2.9.4 Occurrence

Definition	Occurrence represents a situation whereby a BeliefAgent lacks confidence in occurrence existing in a BeliefStatement.
Features	(inherited from Uncertainty parent)
Semantics	There are no further semantics associated with this concept.
Constraints	None.
General Example	"The probability that it will rain tomorrow in Oslo is 30%".
CPS Example	"The probability that packet loss during a videoconference is less than 3.2% is 85%".

3.2.9.5 Time

Definition	Time represents a situation whereby a BeliefAgent lacks confidence in time existing in a BeliefStatement.
Features	(inherited from Uncertainty parent)
Semantics	There are no further semantics associated with this concept.
Constraints	None.
General Example	"It will rain in the morning or afternoon tomorrow".
CPS Example	"The time that VCS spends executing the ramp command of camera is less than 0.1s".

3.2.10 Measure

Figure 8 shows a conceptual model for the Uncertainty Measuring inspired from the concepts reported in [12, 14, 15]. Notice that the measurement concepts presented are by no means complete. Depending on the type of uncertainty a variety of measurements could be used. The purpose of this section is to give a rough idea of commonly known uncertainty measurements.

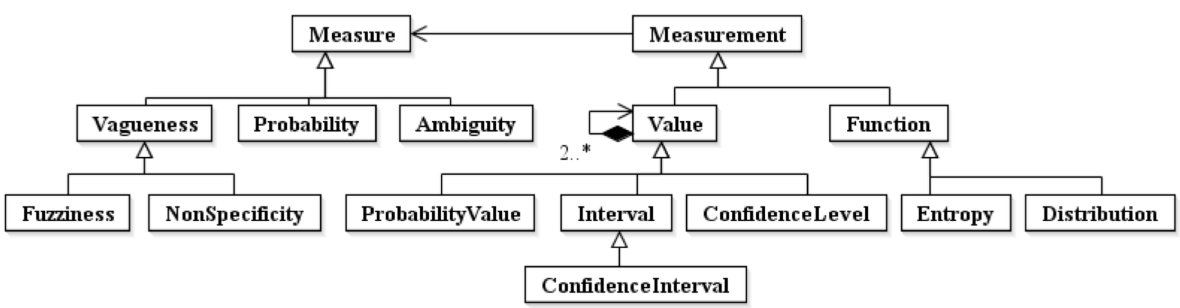

Figure 8. The Measurement of Uncertainty

3.2.10.1 Ambiguity

Definition	Uncertainty in the BeliefStatement is measured using ambiguity way.
Features	(inherited from Measure parent)
Semantics	An uncertainty may be described ambiguously. For example, in the following statement: " The food is hot", the ambiguity is in the measurement, i.e., the food is either hot in terms of temperature or in terms of spices.
Constraints	None.

General Example	"The food is hot", the ambiguity is in the measurement, i.e., the food is spicy or the high temperature.
CPS Example	"The camera is down", the ambiguity is in the measurement, i.e., the camera is either facing down or disconnected. Interested readers may consult [23] for various measures of Ambiguity.

3.2.10.2 Measurement

Definition	Measurement represents the result of measuring stated in the BeliefStatement related to the existing Measure.
Features	• measure – This value represents the measurement method.
Semantics	See Section 3.1.12.
Constraints	None.
General Example	-
CPS Example	-

3.2.10.3 ConfidenceInterval

Definition	Measuring Uncertainty using a confidence interval.
Features	(inherited from Interval parent)
Semantics	There are no further semantics associated with this concept.

Constraints	• ConfidenceInterval has two atomic values.
	`context Value`
	`inv: self.subValues->size()=2`

General Example	-
CPS Example	-

3.2.10.4 ConfidenceLevel

Definition	Measuring Uncertainty using a confidence level.
Features	(inherited from Value parent)
Semantics	There are no further semantics associated with this concept.

Constraints	• ConfidenceLevel is composited of two atomic values.
	`context Value`
	`inv: self.subValues->size()=2`

General Example	"The probability that it will rain tomorrow in Oslo is 30% with confidence level of 2%". In this example, uncertainty is measured as a probability value 30% and a confidence level 2%, i.e., the probability that it will rain ranges from 28% to 32%.
CPS Example	"The probability that packet loss during a videoconference is 2% with confidence level of 0.5%". In this example, uncertainty is measured as a probability value 2% and a confidence level 0.5%, i.e., the probability of packet loss ranges from 1.5% to 2.5%.

3.2.10.5 Distribution

Definition	Uncertainty expressed using a distribution.
Features	(inherited from Function parent)
Semantics	There are no further semantics associated with this concept.
Constraints	None.
General Example	-
CPS Example	-

3.2.10.6 Entropy

Definition	Uncertainty expressed using entropy. More details on the mathematical foundations of entropy can be found in [23].
Features	(inherited from Function parent)
Semantics	There are no further semantics associated with this concept.
Constraints	None.
General Example	-
CPS Example	-

3.2.10.7 Function

Definition	Uncertainty expressed using function.
Features	(inherited from Measurement parent)
Semantics	There are no further semantics associated with this concept.
Constraints	None.
General Example	-
CPS Example	-

3.2.10.8 Fuzziness

Definition	Uncertainty measured by fuzzy methods. More details can be referred to fuzzy logic literature [23].
Features	(inherited from Vagueness parent)
Semantics	There are no further semantics associated with this concept.
Constraints	None.
General Examples	-
CPS Example	-

3.2.10.9 Interval

Definition	Uncertainty expressed using interval.
Features	(inherited from Value parent)
Semantics	A range of values
Constraints	None.
General Examples	-
CPS Example	-

3.2.10.10 Non-Specificity

Definition	Uncertainty measured using non-specificity methods.
Features	(inherited from Vagueness parent)
Semantics	In certain cases, it may not be possible to measure an uncertainty using quantitative measurements and instead qualitative measurements can be used. Such qualitative measurements are classified under Non-Specificity methods.
Constraints	None.
General Example	“It will rain heavily tomorrow in Oslo”. In this example, uncertainty is measured with non-specificity, i.e., heavily.
CPS Example	“The packet loss is low in Oslo”. In this example, uncertainty is measured with non-specificity, i.e., low.

3.2.10.11 Probability

Definition	Uncertainty measured with the probability.

Features	(inherited from Measure parent)
Semantics	A quantitative way of measuring uncertainty.
Constraints	None.
General Example	-
CPS Example	-

3.2.10.12 ProbabilityValue

Definition	ProbabilityValue represents the actual probability value measuring uncertainty.
Features	(inherited from Value parent)
Semantics	There are no further semantics associated with this concept.

Constraints	• ProbabilityValue is an atomic value.
	`context Value`
	`inv: self.subValues->size()=0`
	• ProbabilityValue should be used in probability measurement method.
	`context Measurement`
	`inv: self. oclIsTypeOf (ProbabilityValue) implies (self.measure->notEmpty()`
	`and self. measure.oclIsTypeOf(Probability))`

General Example	-
CPS Example	-

3.2.10.13 Vagueness

Definition	Uncertainty measured with the vagueness methods.
Features	(inherited from Measure parent)
Semantics	There are no further semantics associated with this concept.
Constraints	None.
General Example	-
CPS Example	-

3.2.10.14 Value

Definition	A value measuring uncertainty.
Features	(inherited from Measurement parent)
Semantics	There are no further semantics associated with this concept.
Constraints	None.
General Examples	-
CPS Example	-

4 Infrastructure Level Uncertainty Domain Model

Figure 9 shows our view of CPS infrastructure.

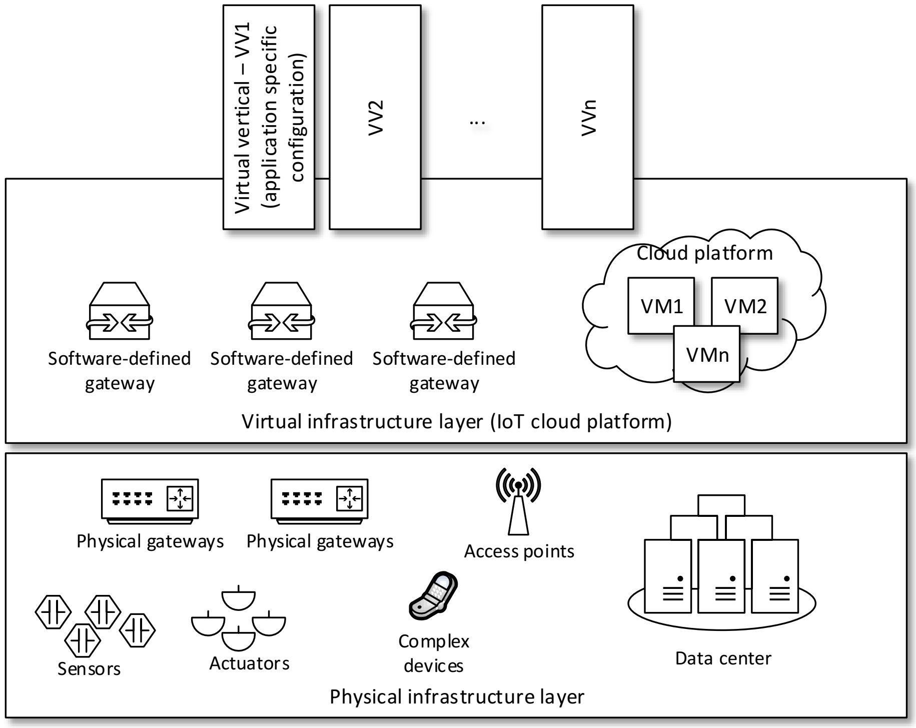

Figure 9. A high-level overview of CPS infrastructure

4.1 Uncertainty states of CPS infrastructure

Generally, errors, faults and uncertain behaviours at the infrastructure level affect the execution of CPS applications independent of the applications business logic. Therefore, classifying the infrastructure level uncertainties can be generic to a large extent, i.e., based on the functionality CPS applications usually expect from such infrastructures to deliver. Some of the responsibilities (functionality) of the CPS infrastructure include:

- Providing communication facilities (i.e., network) among the sensors/actuators and CPS applications/services.
- Providing an execution environment for such applications (e.g., on gateways or in the cloud).
- Providing (temporary and/or permanent) storage for the large amounts of sensory data.
- Providing facilities for generating, pre-processing and delivering sensory data.
- Enabling routing/buffering of actuation requests (from applications to physical actuators).

We mainly focus on uncertainties that affect the aforementioned functionality of the infrastructure. More specifically, such uncertainties affect the expected state of the infrastructure, i.e., the outcome when an application utilizes (e.g., invokes) some of the infrastructure functionality. Generally, such uncertainties can cause the CPS infrastructure to display faulty behaviour (i.e., come into an error state) or some uncertain state (not necessarily an error state).

In the traditional fault, error, failure classifications, faults lead to some form of errors which are manifested as failures at application or service level [24]. The main difference between the traditional (latent) error state and the uncertain state are the causes that lead the CPS infrastructure to transition to such state and in how such state manifests at application level or in the surrounding environment. In our context, uncertainties can coincide with faults, but are much broader category. For example, an empty data channel can be considered as an uncertain infrastructure state, since it can be caused by a sensor failure (error state) or because there is no change in the physical environment, thus nothing is detected by a sensor (normal state).

4.2 Infrastructure level uncertainties properties classes

Our taxonomy classifies the (at design time) known sources of the error and the uncertain states, e.g., the behaviours of system units which are potentially, positively or cumulatively responsible for the error/uncertain states of CPS infrastructure Figure 10 gives an overview of the infrastructure level uncertainties taxonomy for CPS systems. The taxonomy shown in Figure 10 comprises a set of concepts (i.e., uncertainty properties classes), which are a concrete instantiations of the concepts defined in the meta model described in Section 3.We have identified 7 main uncertainty properties classes at the infrastructure level. Next we describe these property classes in more detail and note how these property classes relate to the *U-Taxonomy*'s core model (described in Section 3).

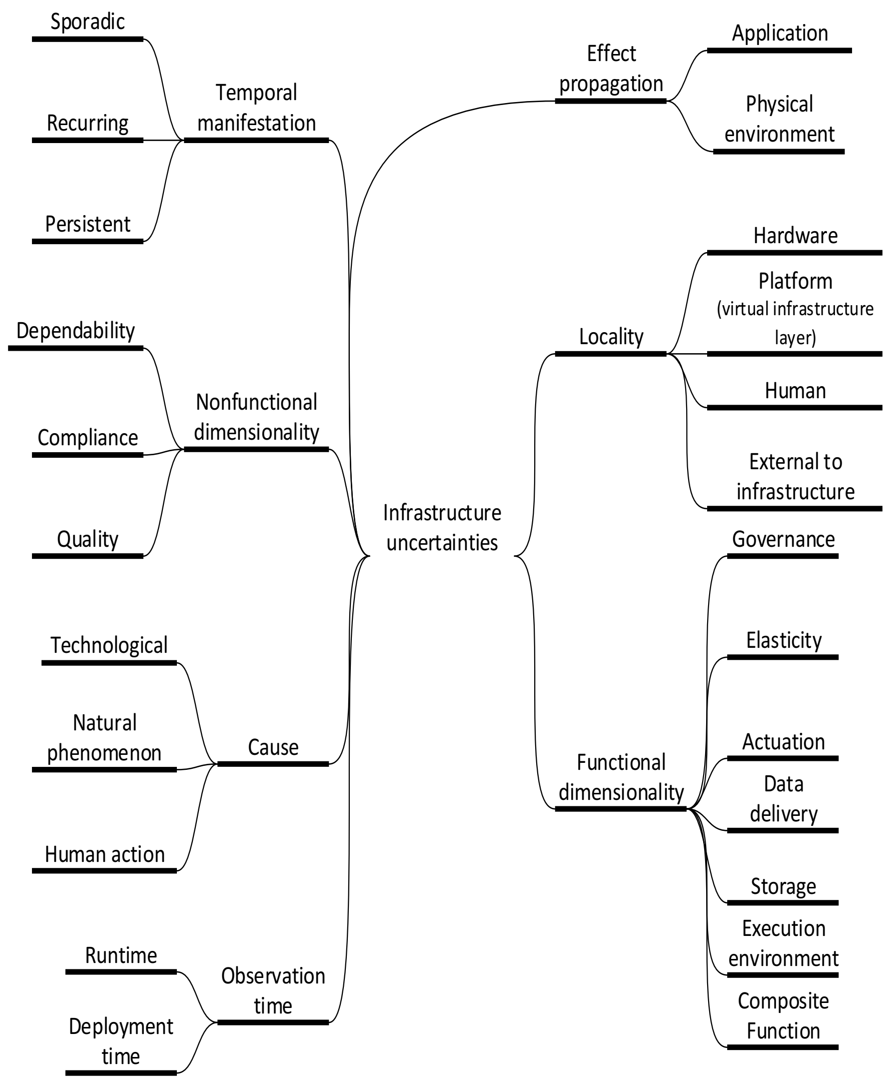

Figure 10. Infrastructure level uncertainties taxonomy for CPS systems

4.2.1 Effect propagation uncertainties (What the uncertainties affect)

Uncertainties at the CPS infrastructure can manifest themselves as failures or as functionality degradation at application level (e.g., [25]) or in the physical environment [26]. For example, empty data channel will obviously be noticed by an application, while malfunctioning chiller wing will be noticed in the physical environment. These uncertainty properties are derived from the Locality and the Effect concepts, introduced in the core model of *U-Taxonomy* (shown in Figure 4)

4.2.2 Uncertainty locality (Where uncertainties occur)

Depending on the locality of the uncertainties occurrence, we differentiate between the uncertainties that are present in the infrastructure itself, i.e., in hardware (e.g., sensors, actuators, gateways, etc.), CPS platform/virtual part of the infrastructure (e.g., cloud services or elasticity controllers), humans interacting with the infrastructure (e.g., for configuring infrastructure functions) and the uncertainties that occur outside the infrastructure and affect the infrastructure, e.g., smoke interfering with normal operation of surveillance cameras. These uncertainty properties are based on the previous work on fault localization [27, 28] and root cause analysis [29]. Further subclasses of Hardware and Platform are CPS Edge, CPS Data Center, and CPS Intermediate Connectivity, describing where the hardware and platform are in the edge of the CPS (sensors, gateways, actuators, etc. are in the physical site), in the data center (cloud services), and in the intermediate network between the edge and the data center. In the core uncertainty model (shown in Figure 4), this corresponds with the Uncertainty Locality concept. A special note is about humans: at the moment we do not consider humans as a part of the infrastructure. However, as humans might affect the infrastructure, humans are considered as an uncertainty locality.

4.2.3 Non-functional dimensionality (Which non-functional property they affect)

The uncertainties can affect the dependability [30] (e.g., safety, availability, reliability, security, etc.), data quality or legal/compliance of the CPS infrastructure [31-33]. It is worth noticing here that the non-functional dimensionality can be used to measure the degree of sensitivity to an uncertainty, where a complete functionality failure is the highest degree and no functionality degradation (e.g., no availability degradation) is the lowest degree. The non-functional dimensionality of infrastructure uncertainties is derived from the core model's general concepts: the Effect and the Measurement owned by Effect (shown in Figure 4).

4.2.4 Causes of uncertainty (What causes them)

The uncertainties can be caused by some natural phenomenon in the surrounding environment, they can be a consequence of human actions or they can be technology caused uncertainties. Under uncertainties with technological cause, we classify all the uncertainties that are caused by some infrastructure phenomenon, which is beyond application developer's control. For example, these can be infrastructure hardware failures or bugs in the virtual infrastructure. Generally, these uncertainty properties represent the phenomenological cause of an uncertainty [34]. The Uncertainty Cause is an instantiation of Cause class from the core model (shown in Figure 4).

4.2.5 Temporal manifestation (How they manifest in time)

The uncertainties can manifest in time as persistent, sporadic or as recurring. Generally, temporal manifestation denotes the duration of the infrastructure uncertainty state caused by that uncertainty. For example persistent uncertainties will cause permanent uncertainty state, i.e., until an outside action (e.g., human intervention) causes the infrastructure to return from the uncertain state to a normal state. These properties are inspired by the traditional software bugs classifications [35]. These concepts are instantiations of Time class from the core model (shown in Figure 7 and Figure 5).

4.2.6 Functional dimensionality (Which functional properties they affect)

As already discussed at the beginning of this section, CPS infrastructure is responsible to provide a specific functionality to the applications. Depending on what functionality class they affect, we differentiate among elasticity, governance, actuation, data delivery, storage, execution environment, or composite function uncertainties. The functional dimensionality is derived from state-of-the-art in CPS infrastructure research [36-44].

4.2.7 Observation time (When do they manifest/become active)

Depending on when in the application lifecycle an uncertainty becomes active, i.e., potentially manifests itself as a failure, we have deployment time or runtime uncertainties. These concepts are instantiations of the Uncertainty Lifetime class form the core model shown in Figure 4.

4.3 Elementary uncertainties families

When classifying the uncertainties, we notice that not all the combination of the uncertainty properties are allowed. For example, it makes no sense to have a natural phenomenon uncertainty which occurs at the platform level (in software). Subsequently we identify the uncertainty families that are most common in practice. The uncertainty families are the permissible combinations of uncertainty properties (without claim of completeness). The families are mainly categorized depending on the functional dimensionality of the uncertainties.

4.3.1 Data delivery uncertainties family

The data delivery uncertainties family includes such uncertainties that affect the infrastructure's facilities for generating, pre-processing and delivering (sensory) data. It includes three main elementary categories: Uncertainties affecting the *dependability of the data delivery* facilities, uncertainties affecting the *quality of data* and uncertainties related to *compliance.*

Name:	Data delivery dependability uncertainties
Definition:	These uncertainties affect the general dependability of the data delivery facilities. They can originate due to a human action or have a technological cause. They are located in hardware or platform. They can have any temporal manifestation and can be observed at any phase of application lifecycle.
Example:	See Figure 11[8]

[8] Note on uncertainties family examples: The uncertainty instances (examples) are classified in a tree structure. The root tree node denotes the uncertainties family and the intermediate tree nodes represent the uncertainty properties classes from the aforementioned infrastructure uncertainties taxonomy. The leaf nodes represent concrete uncertainty instances (examples). The edges are meant to represent logical "and" binding between the uncertainty properties classes.

Name:	Data quality uncertainties
Definition:	These uncertainties affect the quality of the data generated and/or delivered by the CPS infrastructure. They can have a technological cause or originate due to a human action or some natural phenomenon. They can have any defined locality. They can have any of the defined temporal manifestations and can be observed at any phase of application lifecycle.
Example:	See Figure 11

Name:	Data delivery legal/compliance uncertainties
Definition:	These uncertainties affect the legal or compliance aspects of the data delivery process. They originate due to human actions, external to infrastructure. They are persistent and observed during application's runtime.
Example:	See Figure 11

4.3.2 Actuation uncertainties family

The actuation uncertainties family includes such uncertainties that affect the infrastructure's mechanisms related to routing, buffering, delivering and ordering (e.g., by priorities) of actuation requests that originate on the application level and are propagated to the physical or virtual actuators. All uncertainties from this family are observed during runtime. The actuation uncertainties family comprises three main elementary categories: Actuation legal/compliance uncertainties, actuation uncertainties affecting the *dependability of specific applications* and actuation uncertainties affecting the *dependability of environment.*

Name:	Actuation legal/compliance uncertainties
Definition:	These uncertainties affect the legal or compliance aspects of the actuation process. They are caused by human actions, in the platform and are mainly persistent uncertainties.
Example:	See Figure 12

Name:	Actuation dependability uncertainties in applications
Definition:	These uncertainties affect the general dependability of the applications, i.e., actuation facilities. They can be caused by a human action or technology. They are located in hardware or platform. They can have any temporal manifestation defined in the taxonomy.
Example:	See Figure 12

Name:	Actuation dependability uncertainties in environment
Definition:	These uncertainties affect the general dependability of the physical environment. They can have any origin specified in the taxonomy. They usually located in the hardware or external to the infrastructure and have any of the specified temporal manifestations.
Example:	See Figure 12

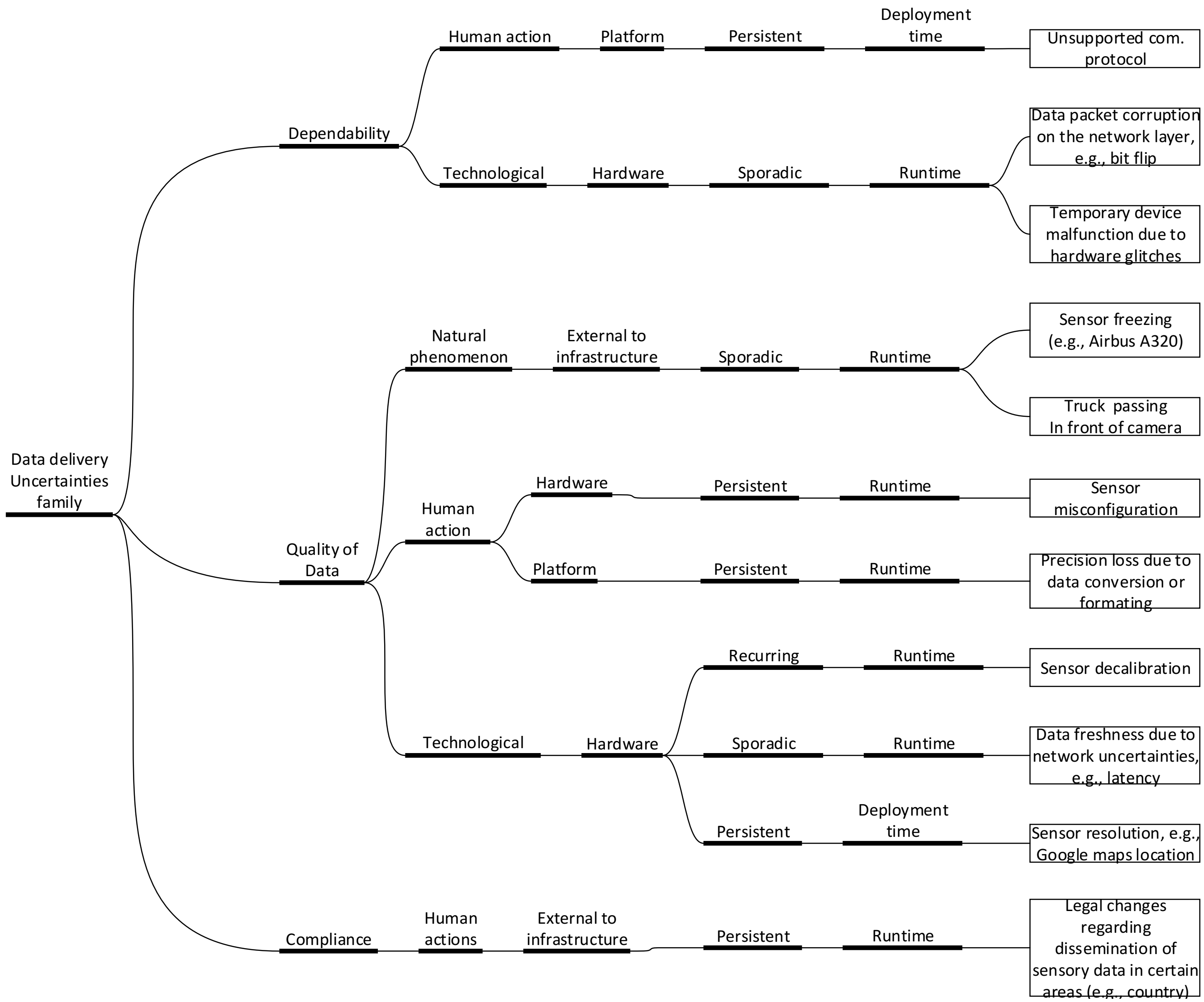

Figure 11. Data delivery uncertainties family

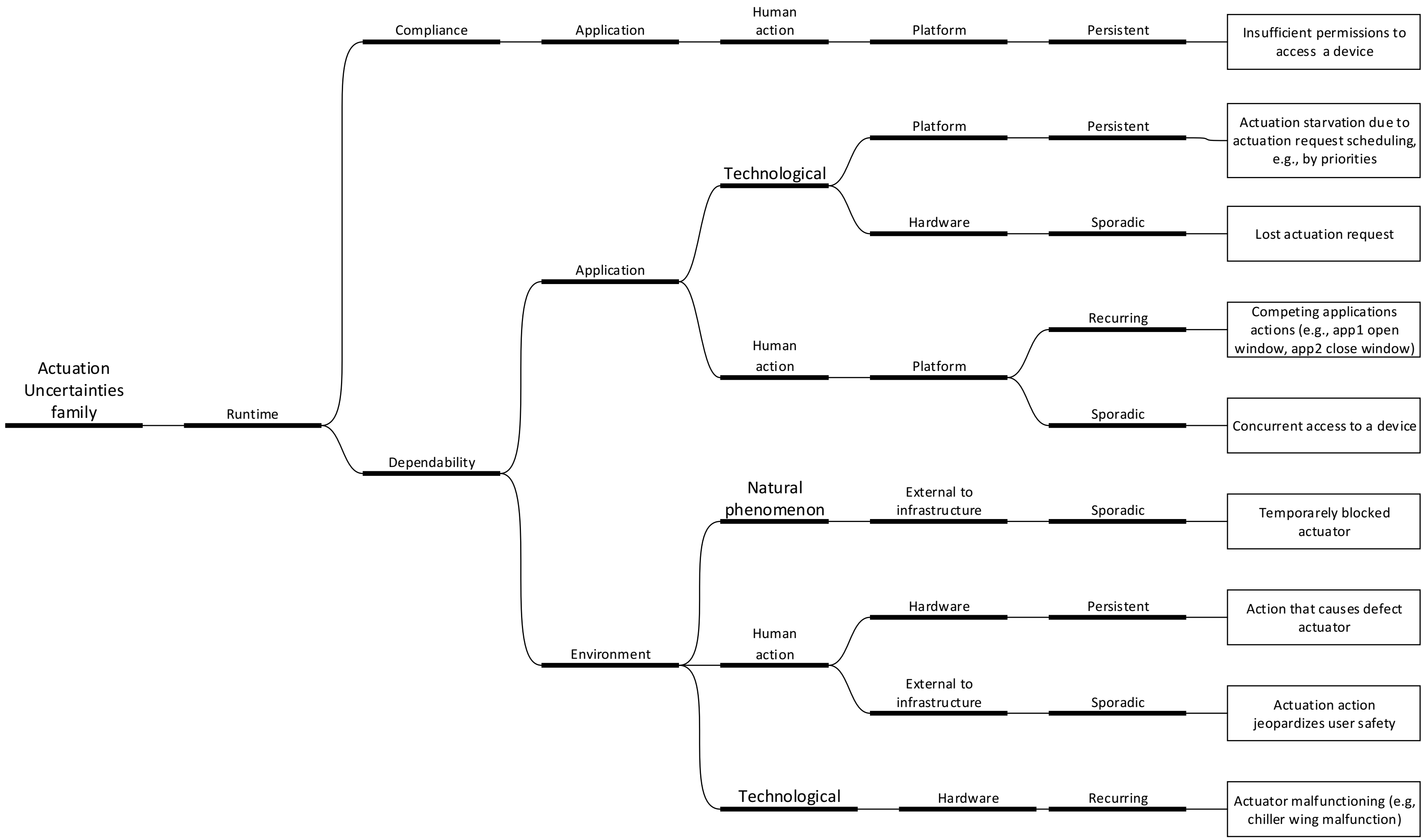

Figure 12. Actuation uncertainties family

4.3.3 Execution environment uncertainties family

The execution environment uncertainties family comprises uncertainties about the assumptions made by application developers about the underlying infrastructure functionality. They interfere with the infrastructure's ability to support application execution, thus are classified as application uncertainties. The execution environment uncertainties family comprises two main elementary categories: Execution environment uncertainties *observed at application deployment* and execution environment uncertainties *observed at application runtime.* However, such uncertainties might be also observed outside the application runtime.

Name:	Deployment time execution environment uncertainties
Definition:	These uncertainties are observed during application's deployment phase. The non-functional dimensionality of such uncertainties is either dependability or legal/compliance and their locality manifestation is mostly at hardware or platform level. They have a technological origin or can be caused by human actions. They can have any of the defined temporal manifestations.
Example:	See Figure 13

Name:	Runtime time execution environment uncertainties
Definition:	These uncertainties interfere with application's execution, thus are observed during its runtime, mainly by affecting infrastructure's dependability at hardware of platform level. They can have any of the defined temporal manifestations or origin.
Example:	See Figure 13

4.3.4 Storage uncertainties family

The storage uncertainties family includes uncertainties that affect the infrastructure's facilities for persistent storage of monitoring (sensory) data. This family mainly manifests as failure at application level when such applications perform batch data analytics (As opposed to the data delivery facilities, where the focus is on real-time data processing). All uncertainties from this family are observed during application runtime. The storage uncertainties family comprises three main elementary categories: Uncertainties affecting the *dependability of the storage* facilities, uncertainties affecting the *quality of the historical data* and uncertainties related to *legal/compliance regulating sensory data storage*.

Name:	Storage quality uncertainties
Definition:	These uncertainties affect the quality of the data (most notably historical sensory data) stored in the CPS infrastructure. They can have a technological origin or are caused by a human action at hardware or platform level. They can have any of the temporal manifestations specified in the taxonomy.
Example:	See Figure 14

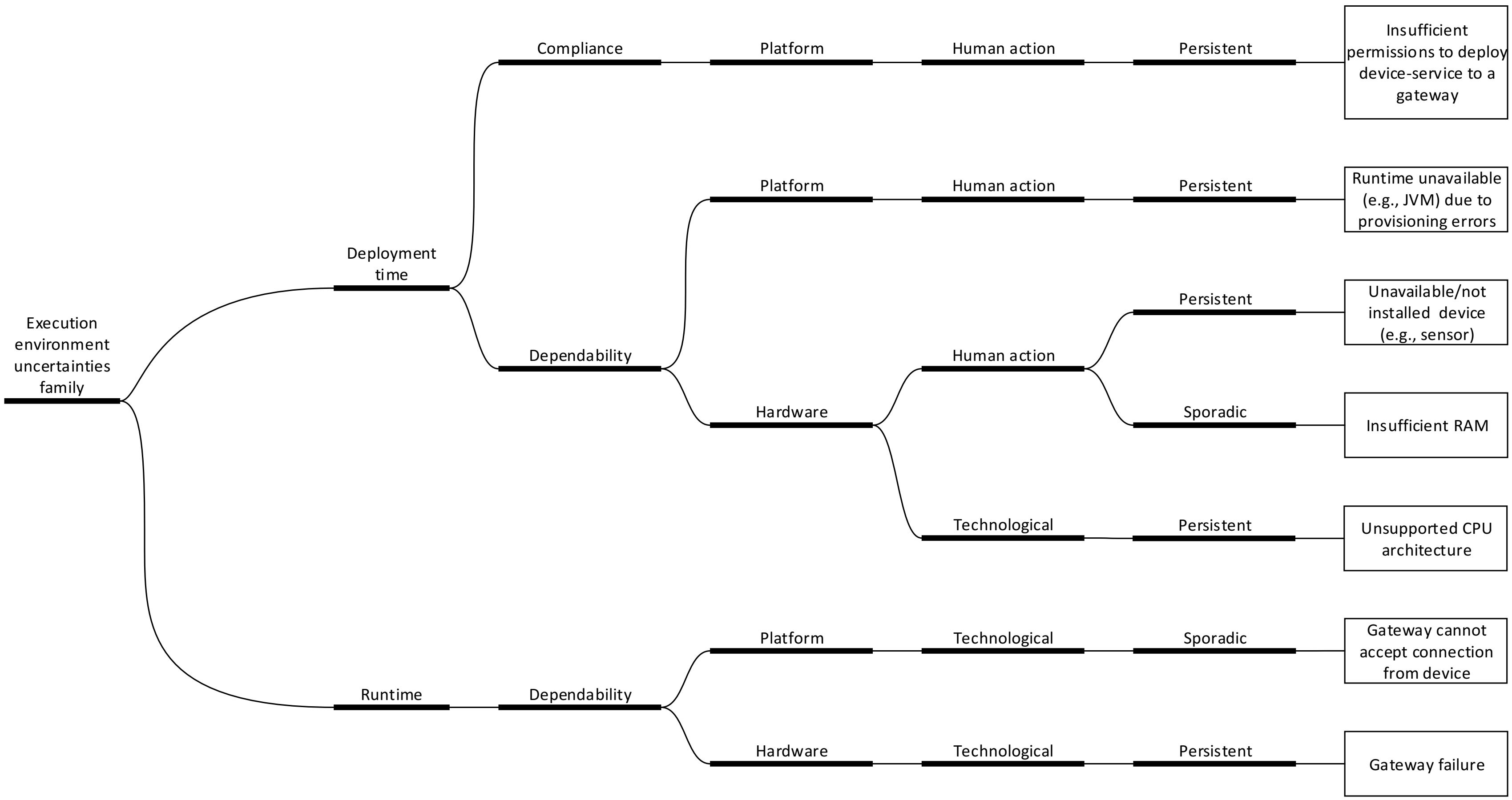

Figure 13. Execution environment uncertainties family.

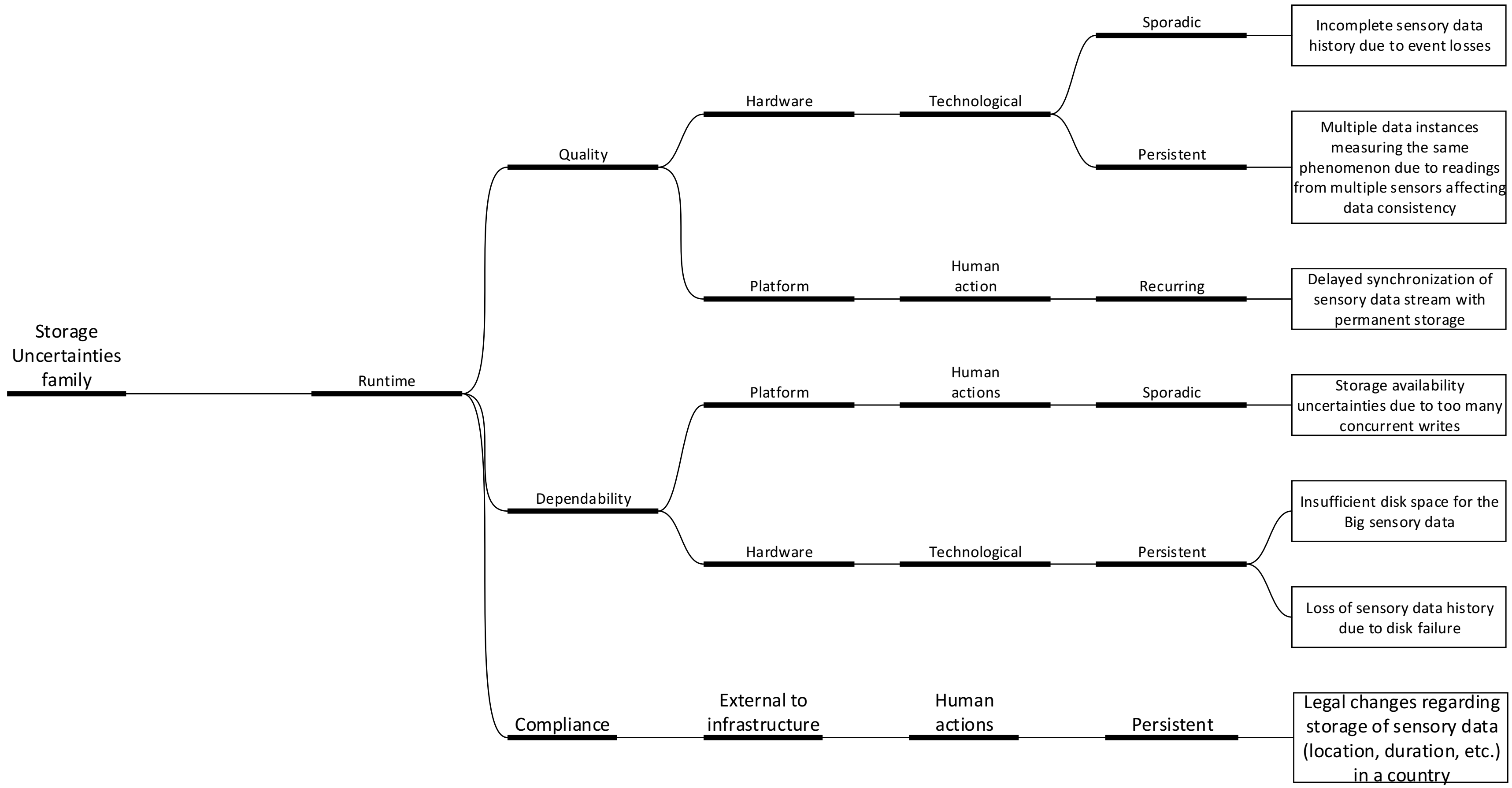

Figure 14. Storage uncertainties family.

Name:	Storage dependability uncertainties
Definition:	These uncertainties affect the general dependability of the data storage facilities. They can have a technological origin or are caused by human action. They are located in hardware or platform and can have any temporal manifestation.
Example:	See Figure 14

Name:	Storage legal/compliance uncertainties
Definition:	These uncertainties affect the legal or compliance aspects related to the data storage. They originate due to human actions, external to infrastructure and are persistent uncertainties.
Example:	See Figure 14

4.3.5 Elementary uncertainties families – aggregated view

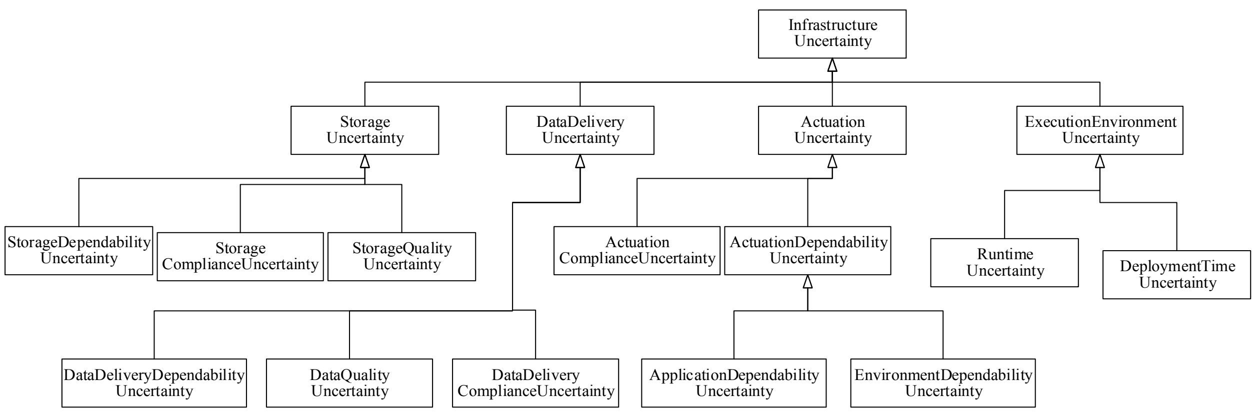

Figure 15. UML diagram showing the elementary uncertainties families

4.4 Composite uncertainties families

Composite uncertainties appear mostly in infrastructure's higher-level functionality – most notably, but not limited to governance and elasticity facilities, thus they mostly manifest at the higher levels in the infrastructure stack, e.g., the infrastructure software platform. Composite uncertainties mostly come into effect through the uncertainties propagation and/or uncertainties aggregation from the elementary uncertainties families (described in Section 1.3). It is also worth noticing that composite uncertainties can be used as an extension point of the infrastructure uncertainties classification.

4.4.1 Governance uncertainties family

The governance uncertainties family includes uncertainties that affect the infrastructure's facilities responsible to realize CPS governance processes or the uncertainties which make such processes invalid.

Name:	Governance process execution uncertainties
Definition:	Governance process execution uncertainties affect the dependability of the governance process during runtime. They are observed at applications runtime and are mainly located in the platform. They usually have a synthetic origin

	and any permissible temporal manifestation.
Example:	For example a golf course management application polls diagnostic data from vehicles (e.g., with CoAP). However, a golf course manager could design a governance process that is triggered in specific situations such as in case of emergency. Such process could, for example, increase the update rate of the vehicle sensors and change the communication protocol to MQTT in order to satisfy a high-level governance objective, e.g., company's compliance policy to handle emergency updates in (near) real-time. In this context it is uncertain whether the governance process will be executed consistently across the infrastructure, because some vehicle sensors might not support functionality to dynamically change their update rate.

Figure 16 shows UML diagram of the composed uncertainties families related to governance uncertainties.

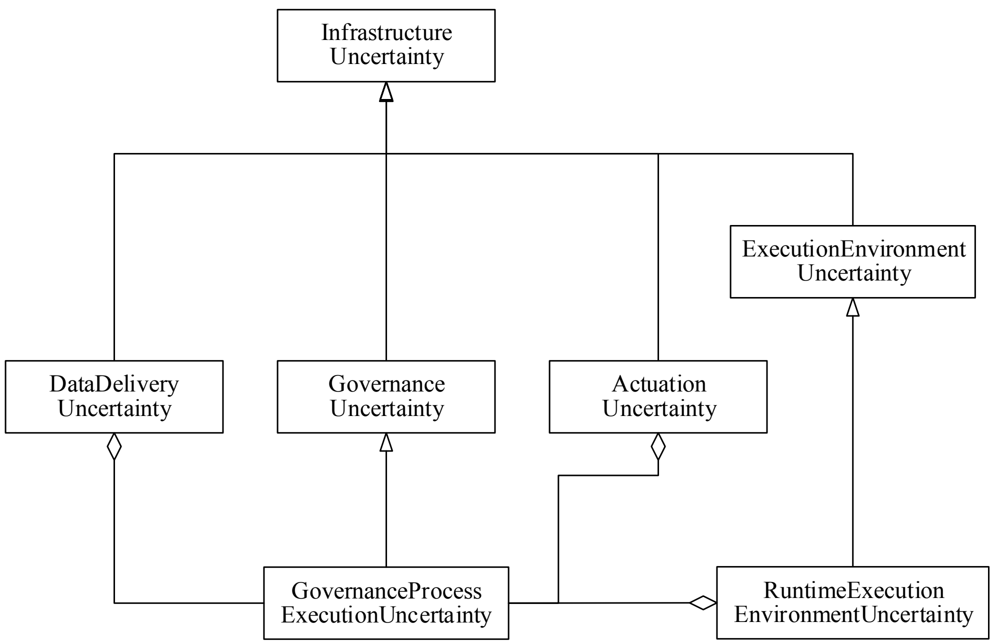

Figure 16. Governance uncertainties families (partial view)

4.4.2 Elasticity uncertainties family

Elasticity is dependent on multiple factors. First, elasticity decisions are taken based on monitoring information, so uncertainty related to monitoring has great importance. Based on monitoring information, elasticity decisions are enforced through a combination of software and hardware actuation mechanisms, each of them also potentially introducing their own uncertainties.

Name:	Monitoring data uncertainties
Definition:	These uncertainties affect elasticity of the system, and can refer to uncertainty of monitoring data quality, e.g., availability or freshness. They usually have a synthetic origin, i.e., required information is not collected and monitored due to a software error. Another cause can be software failure of monitoring system, or of monitoring information data source. Another cause is data

	collection mechanism and intervals, especially considering poll-based data collection systems, which collect and report monitoring information only at certain time intervals. They are located in platform. They can have any temporal manifestation and can are observed at application runtime.
Example:	1. Monitoring layer is poll-based, and collects monitoring information every 5 seconds, but reports it only every 10 seconds. Thus, when examining monitoring information, we retrieve information which can be up to 15 seconds old. Thus, it is uncertain if the old information still accurately represents the current behaviour of the application. 2. Information data source (e.g., a Web Server reporting response time), is overloaded or crashes, and does not report data anymore. This can generate two problems, depending on the behaviour of the monitoring layer, each equally severe: a. The first problem is if the monitoring layer uses the last returned value as the current one and continuously returns it to anyone requesting monitoring information. This leads to false data being produced by the monitoring layer. b. The second issue is if the monitoring layer just ignores the missing data, leading to application behaviour information not being available.

Name:	Cloud Service behavioural uncertainty after actuation
Definition:	These uncertainties affect the elasticity of the application by reducing the effectiveness, or affecting the impact of enforced elasticity actions. They originate in the (cloud provider's) platform not offering consistent performance across different instances of the same used cloud service, either to colocation or congestion or virtual resources, or complete/partial failure due to underlying cloud software and hardware infrastructure.
Example:	1. Two instances of a Virtual Machine or Virtual Network services promising a certain performance might provide different maximum I/Ops and respectively Bandwidth, depending on how the cloud provider distributes the load from the virtual resources through the underlying physical infrastructure. 2. Instances of cloud services can fail during their runtime, due to unforeseen and usually hidden reasons, such as bugs in the software or hardware used by the cloud provider.

Figure 17 shows UML diagram of the composed uncertainties families related to elasticity uncertainties.

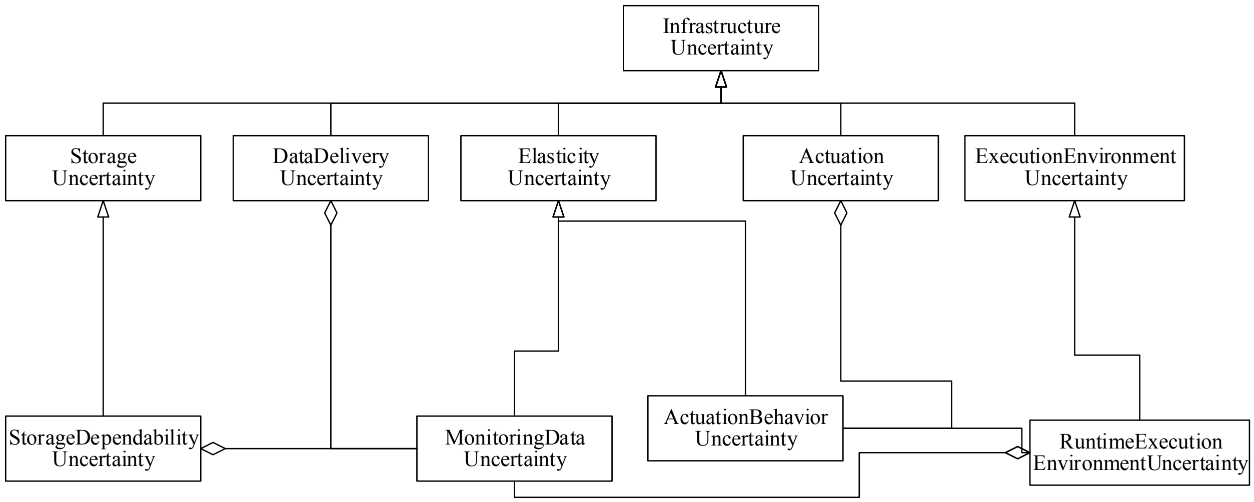

Figure 17. Elasticity uncertainties families (partial view)

4.4.3 Unknown uncertainties at infrastructure level

Although unknown uncertainties are out-of-scope of this task, we notice that a very large number of such uncertainties can manifest themselves at the infrastructure level. This is mainly due to complex dependencies among the infrastructure components and effects of uncertainty propagation and/or uncertainty aggregation between such components. Generally, the root cause, locality, temporal manifestation, etc., of unknown uncertainties are inherently difficult if not impossible to determine. Therefore, classification of such uncertainties is usually application specific and can be classified under different or even multiple elementary classes depending on the task-at-hand.

5 Application Level Uncertainty Domain Model

Uncertainties at the application level come from the environment and are accordingly called environmental uncertainties [45-49]. According to Cheng [46], environmental uncertainties come from the physical environment and the cyber environment. Uncertainties from the physical environment come from unforeseen or environmental conditions with a lack of knowledge about it and may also result from sensor failures or noisy environments [49]. Uncertainties from the cyber environment may result from malicious threats or unexpected (human) input [49].

This section describes properties specific for uncertainties at the application level based on [46-59]. It supplements the model from the integration level with concepts specific for application level uncertainties.

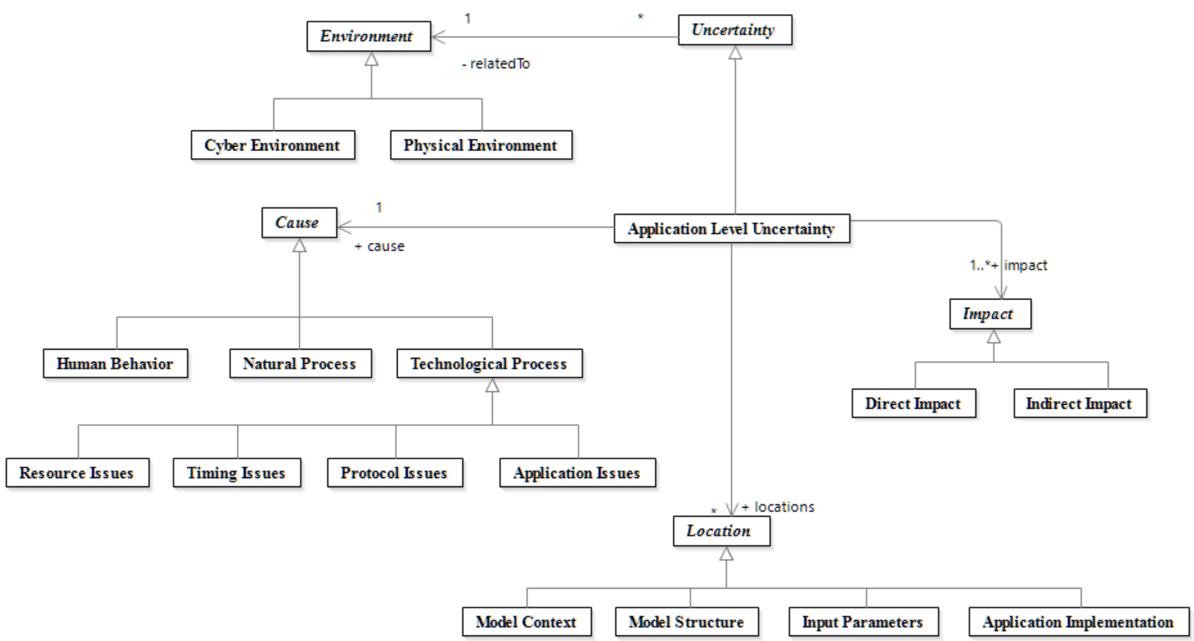

Figure 18. Conceptual Model of Uncertainties at the Application Level

5.1 Uncertainty Nature

The nature of an uncertainty depends on whether the knowledge with respect to an uncertainty is incomplete or whether it is results from an inherently or variable phenomenon [60].

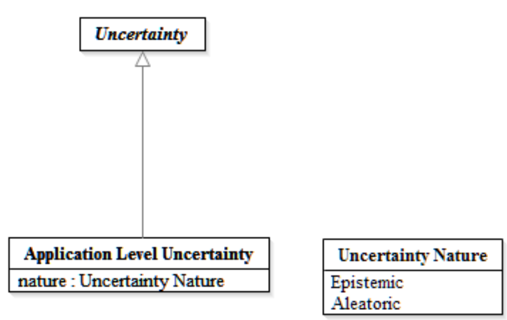

Figure 19. Nature of Uncertainties

The metamodel uses the classical distinction between:

- *Epistemic uncertainty* results from incomplete knowledge, unreliable or imperfect data or even a process that is insufficient in order to build knowledge from acquired data [60]. Such uncertainty "may be reduced by more research and empirical efforts" [58]. This can be expressed with the concept of Knowledge about Uncertainty shown in Figure 3.
- *"Aleatory uncertainty* due to inherent variability of the some parts under consideration or randomness of events" [60]. This is denoted as variability uncertainty by Walker [58]. This concept is related to Random concept in the integration level uncertainty shown in Figure 5.

5.2 Location

The location of uncertainty refers to the place where the uncertainty is located within a model [58, 59].

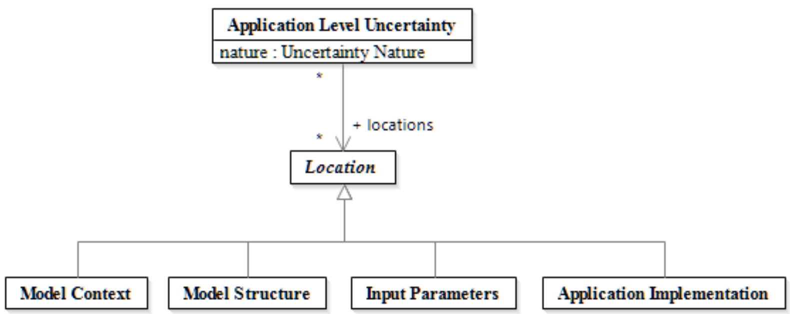

Figure 20. Location Conceptual Model

An uncertainty can be located in the parts of a model described in the following. This concept is related to Locality concept in the integration level uncertainty shown in Figure 4.

5.2.1 Model Context

The context of the model refers to the identification of the boundaries of the model, i.e., the details of the real world that are contained in the model [58]. This is a matter of abstraction from the real world. The model may be too abstract or too concrete for the desired purpose.

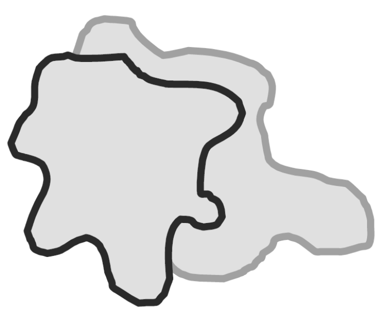

Figure 21. Model Context Uncertainty – "Ambiguity in the definition of the boundaries of the system" [58]

5.2.2 Model Structure

Model structure uncertainty concerns the form of the model [58]. This uncertainty refers to how accurately the structure of the model represents the subset of the real world that has to be modeled, including system behavior and relationships between model elements [58].

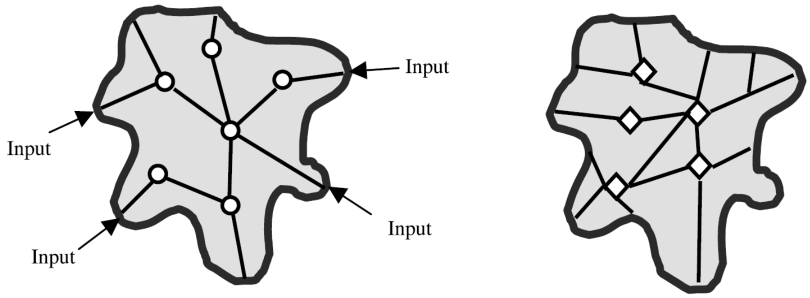

Figure 22. Dominant relationships (on the left) and a different interpretation (on the right) representing model structure uncertainty [58]

5.2.3 Input Parameters

Input Parameters uncertainty is often identified as parameter uncertainty. Such uncertainties are associated with the actual values of variables given as input to the model and with the "methods used to calibrate the model parameters" [58]. This is associated with as Data Uncertainty as shown in Figure 7.

5.2.4 Application Implementation

Application Implementation means that an uncertainty is neither located the input parameters, model context nor model structure but in the implementation of the application.

5.3 Environment

As discussed in Section 1.2, uncertainties at the application level results from uncertainties in the environment. These can be classified in uncertainties from the physical environment and the cyber environment [46]. It can be expressed by instances of the Environment Uncertainty as shown in Figure 7.

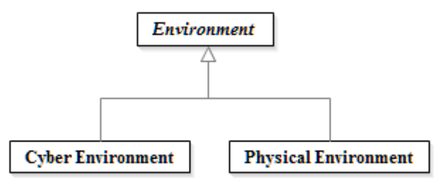

Figure 23. Environment of an Uncertainty

5.3.1 Cyber Environment

Uncertainties from the cyber environment may result from malicious threats or unexpected (human) input [49]. They may also result from the communication with other environmental systems, e.g. other cyber-physical systems.

Example: An adversary may try to get access to a camera of a video conference system without permission.

5.3.2 Physical Environment

Uncertainties from the physical environment come from unforeseen or environmental conditions with a lack of knowledge about it and may result from sensor failures or noisy environments [49].

Example: A video conference system has a camera with autofocus. A person is moving faster than the camera of the system is able to adjust the focus.

5.4 Cause

The cause of an uncertainty at the application level denotes what kind of instance is initiating the uncertainty. This can be expressed by instances of the uncertainties associated with the cause association in the Integration Level as shown in Figure 4.

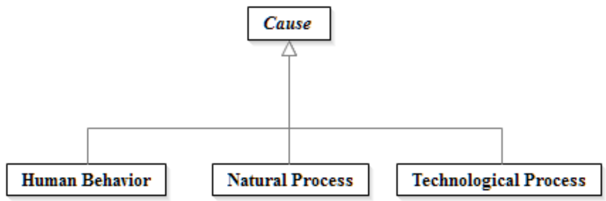

Figure 24. Origins of an Uncertainty

5.4.1 Human Behaviour

An uncertainty may result from a human behavior. A person that regularly interacts with a CPS or an adversarial may show such a behavior.

Example: The examples above denote human behavior that may cause uncertainties at the application level.

5.4.2 Natural Process

A natural process can cause uncertainties.

Example: Solar flares may have an impact on radio communication if it happens with a certain strength.

5.4.3 Technological Process

A technological process can cause uncertainties.

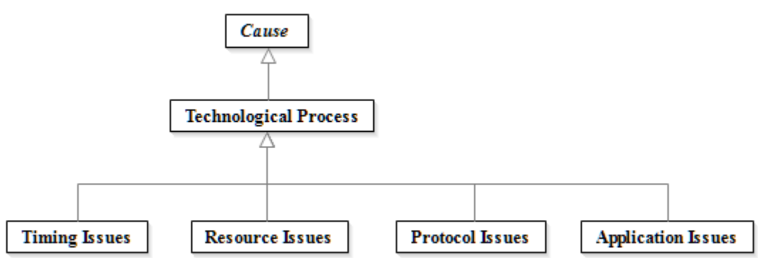

Figure 25. Subclasses of Technological Process

5.4.3.1 Timing Issues

Timing issues result from the uncertainty whether a system is working with the expect performance while abstracting of real time, e.g. by a cycle counter.

5.4.3.2 Resource Issues

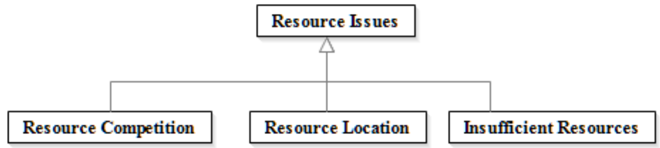

Figure 26. Subclasses of Resource Issues

Resource Issues are reflecting different issues with respect to the cyber world as well as to the real world:

- Resource Competition means that two instances are working on or using the same resources and thus interfering with each other.
- Resource Location means that the expected resource is not where it is expected t be.
- Insufficient Resources comprises uncertainties that result regarding the demanded resources and the expected resources where the demanded resources are higher than the expected resources, e.g. a missing resource item in the real world or insufficient CPU resources with respect to the cyber world.

5.4.3.3 Protocol Issues

Protocol issues summarize different uncertainties with respect to communication protocols.

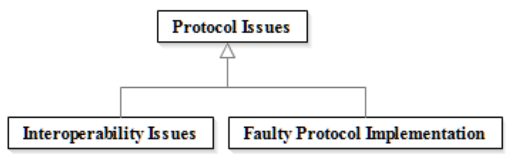

Figure 27. Subclasses of Protocol Issues

5.4.3.3.1 Interoperability Issues

Interoperability Issues occur if the specification of a communication protocol is ambiguous and two communication partners differ in their protocol implementation with respect to ambiguous specification items. This is not a protocol implementation error but a differing interpretation of an ambiguous protocol specification.

5.4.3.3.2 Faulty Protocol Implementation

Faulty Protocol Implementation is a result of an incorrect protocol implementation leading to communication errors between two communication partners, e.g. different components of the application or between the application and the infrastructure of the cyber-physical system.

Example: The communication with another system that stops working because it does not reply to a request

5.4.3.4 Application Issues

Application Issues is comprises uncertainties inherent to the application itself.

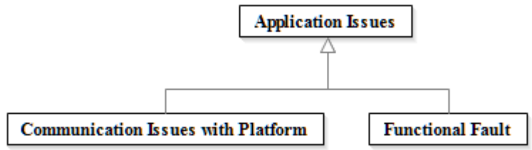

Figure 28. Subclasses of Application Issues

- Communication Issues with Platform is referring to situations where the application fails to communicate with platform devices, maybe resulting from a bad application configuration or other issues.
- Functional Faults means traditional implementation bugs within the application.

5.5 Impact

This concept represents the impact of an uncertainty from the environment to the impacted element such as hardware and/or application. This can be expressed by instance of the uncertainties associated with the effect association in the Integration Level as shown in Figure 4.

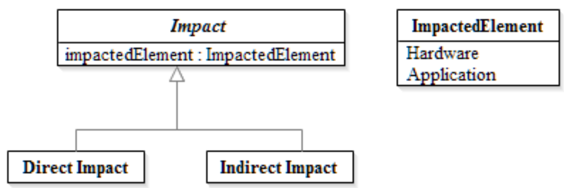

Figure 29. Impact of an Uncertainty on a CPS

5.5.1 Direct Impact

A direct impact resulting from an environmental uncertainty directly influences the physical interface of a CPS, i.e., its sensors or actuators.

Example: Radiation may directly affect the functioning of a sensor.

5.5.2 Indirect Impact

An indirect impact of an environmental uncertainty affects the application logic of the CPS.

Example: A cyber-attack may alter the application logic, e.g., by an SQL injection that changes the database by adding additional users and changing permissions.

5.6 Examples

The section provides different examples of uncertainties at the application level and tries to group them in different families.

5.6.1 Communication Uncertainties

Name:	External Communication Uncertainties
Definition:	Such uncertainties result from unexpected communication issues on protocol level when exchanging data with another system.
Example:	Such uncertainties are usually interoperability issues if the specification is ambiguous, see Figure 30.

Name:	Communication Reliability Uncertainties
Definition:	Those uncertainties result from an unreliable communication medium that leads to interruption in the communication . [52]
Example:	"Notoriously, communication delays are also uncertain, especially when collision-based and wireless protocols are used" [49, 54], see Figure 30.

Name:	Protocol Uncertainties
Definition:	Those uncertainties result from protocol conformance issues and time delays in the communication resulting from faulty implementations of one or more communication partners.
Example:	A communication cut-off resulting from a faulty protocol implementation.

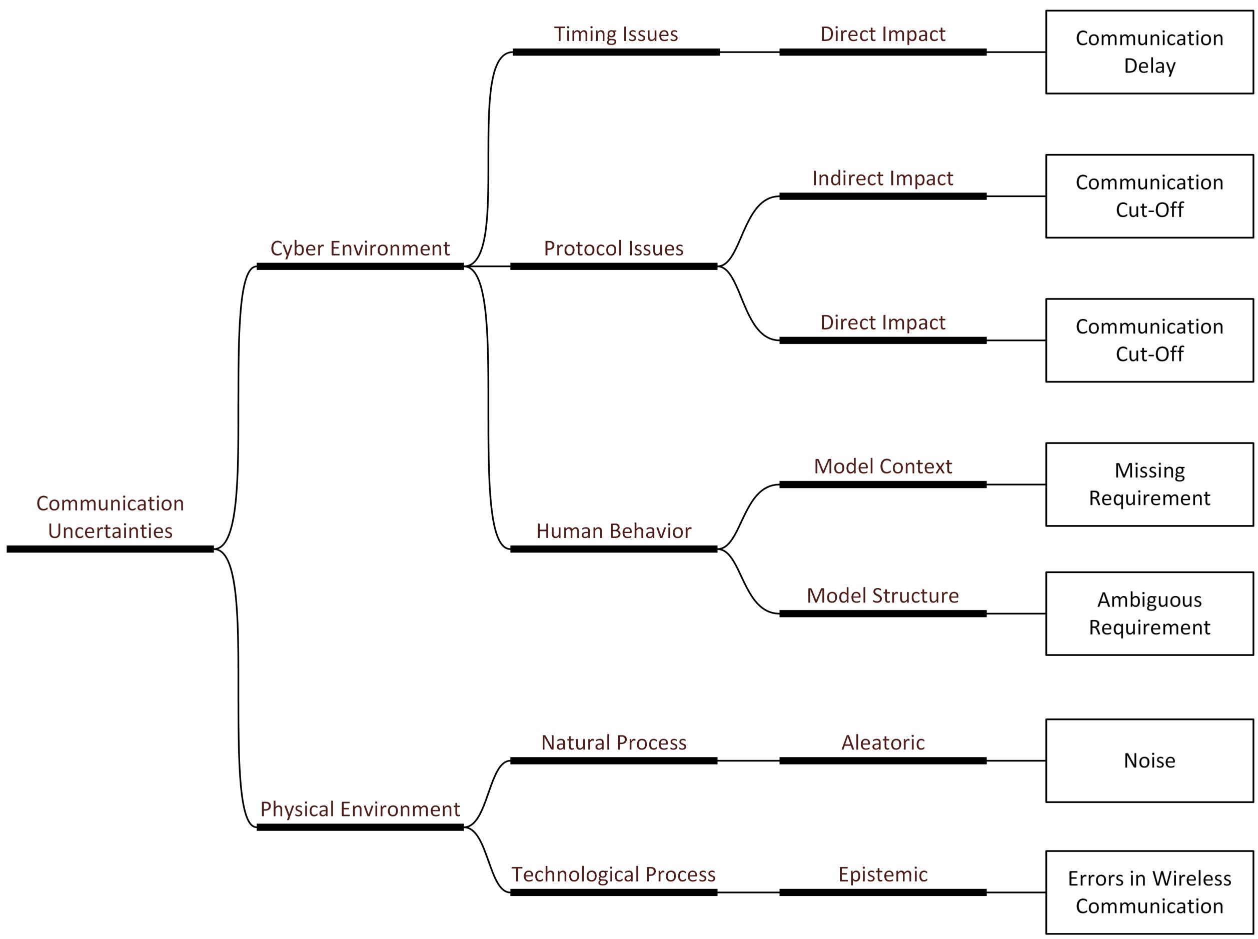

Figure 30. Communication Uncertainty Examples

5.6.2 Attack Uncertainties

Name:	Cyber Attack Uncertainties
Definition:	Such uncertainties result from attacks performed via the cyber environment of a CPS. Cyber-attack uncertainties usually happen via the access through a computer network.
Example:	There are many instances of this family, e.g., different kinds Denial-of-Service attacks, different kinds of injection attacks, e.g., the most famous is SQL injection, see Figure 31.

Name:	Physical Attack Uncertainties
Definition:	Such uncertainties result from attacks performed via the physical environment of a CPS. They may result from uncertainties with indirect interaction with the CPS, e.g., by sensor noise, i.e., jamming, or from uncertainties with direct interaction, e.g. destruction of a sensor, e.g., due to goods the system is interacting with.
Example:	"An attacker does not need to break into the computer to affect such a system, but could cause a coordinated series of physical actions that are sensed and which cause the system to respond in an unexpected manner" [53], see Figure 31.

Name:	Composite Attack Uncertainties
Definition:	Such uncertainties result from attack performed via the cyber environment and/or the physical environment of a CPS. Such an attack is exactly coordinated in order to achieve a certain effect.
Example:	Usually, several vulnerabilities distributed within several components of the attacked systems are exploited. Such an attack is profoundly specific for a certain system.

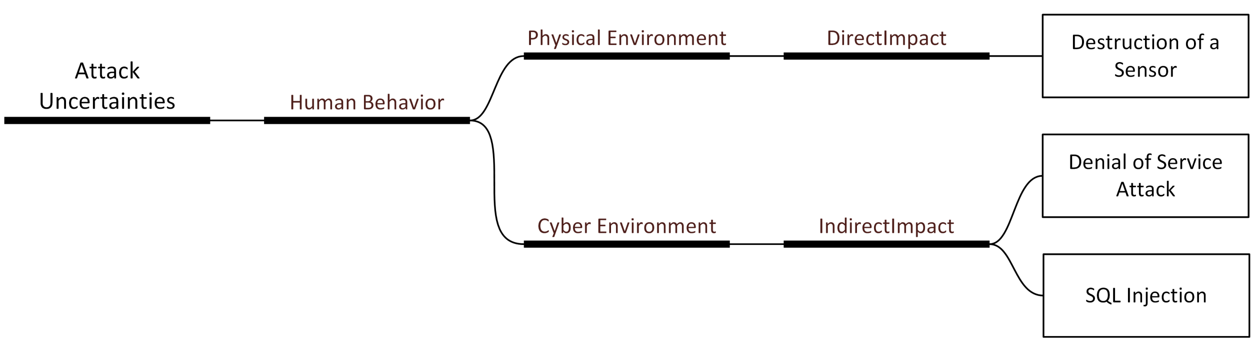

Figure 31. Attack Uncertainties at the Application Level

5.6.3 Application User Behaviour

Name:	Unanticipated Human Interactions [47, 49, 58]
Definition:	This family comprises interaction from the physical environment as well as from the cyber environment.
Example:	A human may interact with an actuator of a system but may behave in an unpredictable way and thus, may interfere with the actuator, see Figure 32.

Name:	Non-Compliant Human Behaviour
Definition:	Such uncertainties result from humans that do not behave according to compliance guidelines.
Example:	E.g. opening unsafe links

Name:	Technical Misuse
Definition:	Such uncertainties result from the misuse of the system.
Example:	A human interacts in an invalid way with the system or inputs wrong or invalid data.

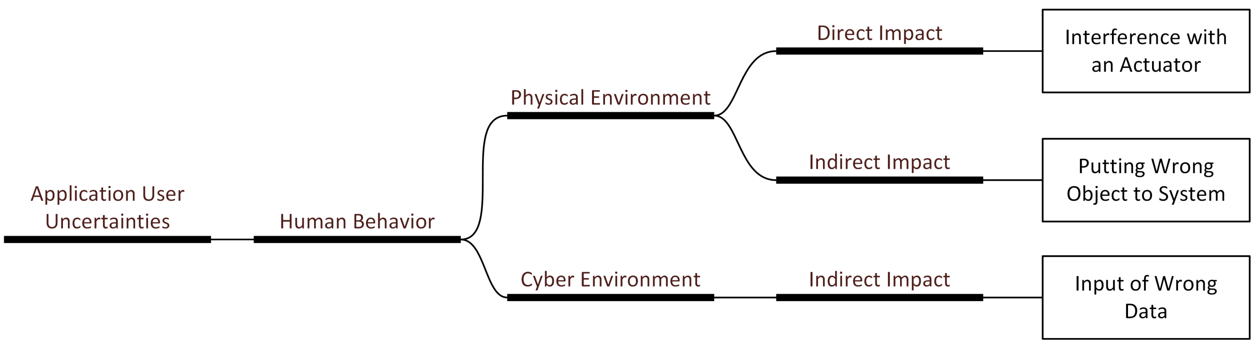

Figure 32. Examples of Uncertainties with respect to the Behavior of the Intended User

5.6.4 Physical Environment Uncertainties

Name:	Environmental Conditions
Definition:	Such uncertainties may result from unforeseen environmental conditions, e.g. weather conditions or radiation. [48]
Example:	"The weather condition that an aircraft will face during a mission is a random parameter, and it becomes a stochastic process for long distance missions." [47] "An adaptive cruise control system was unable to accurately estimate the distance remaining between itself and a vehicle in front of it due to moderate levels of sensor noise across its monitoring infrastructure. As a result, the autonomous, intelligent vehicle system failed to decelerate in time, collided with the vehicle in front, departed from its driving lane temporarily because of the collision, and then continued to collide with the other vehicle in order to re-enter the driving lane." [48] See Figure 33.

Name:	Physical Environmental Interaction Uncertainties
Definition:	The family of this uncertainties results from interaction in the physical environment of a system it interacts with. This may lead to situations the system cannot deal with, e.g. due to fast moving objects or blocked objects.
Example:	See Figure 33.

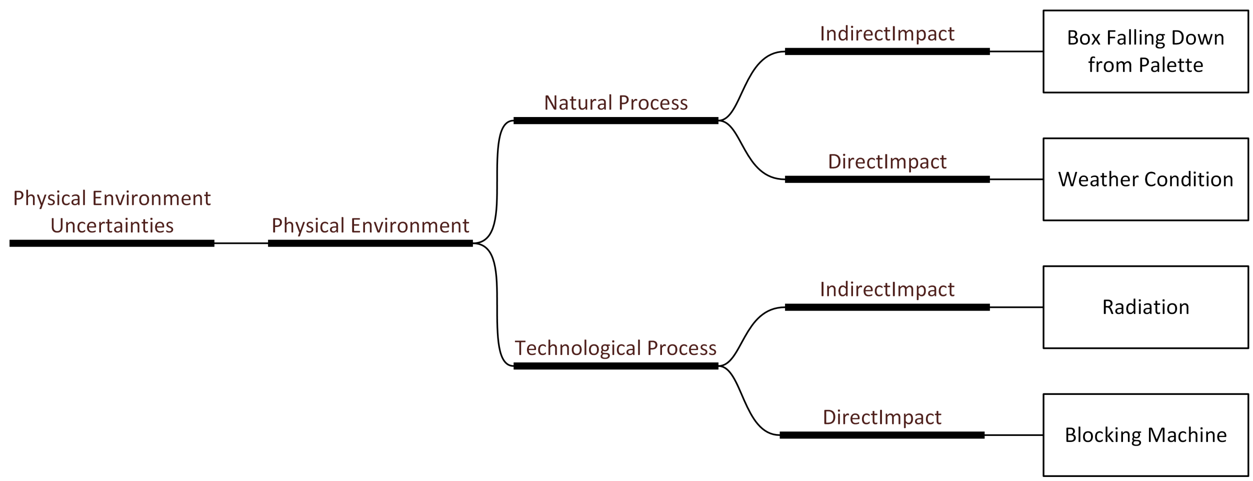

Figure 33. Examples of Uncertainties Resultom from the Physical Environment

5.6.5 Resource Uncertainties

Name:	Resource Uncertainties
Definition:	Resource competition uncertainties result from two application parts are interfering in the usage of physical or technical resources.
Example:	Two processes of the application have a high CPU usage and are running at the same time.

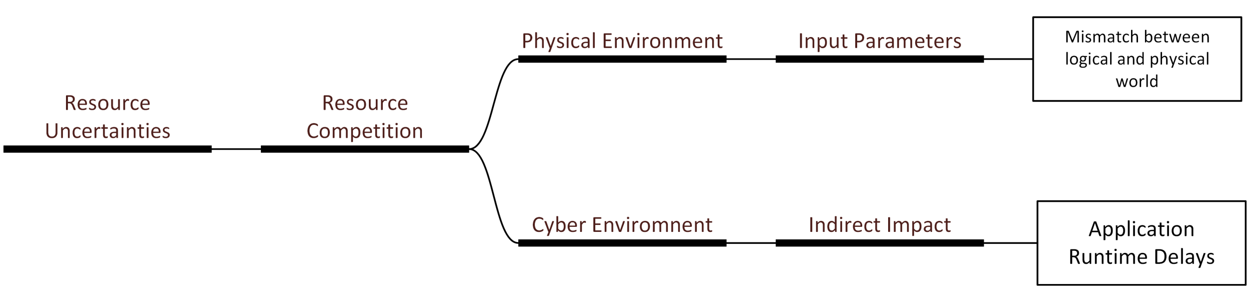

Figure 34: Examples of Resource Uncertainties

6 Related Work

Uncertainty is a term that has been used in various fields such as philosophy, physics, statistics and engineering to conceptually describe a state of having limited knowledge where it is impossible to exactly tell the existing state, a future outcome or more than one possible outcome [61]. Various uncertainty taxonomies have been proposed in literature from different perspectives for various domains. For instance, from an ethics perspective, uncertainties are classified as objective uncertainty and subjective uncertainty, both of which are further classified into subcategories to support decision-making [7]. In health care, uncertainty has often been defined as "the inability to determine the meaning of illness-related events" [8] and comprehensive domain-specific uncertainty taxonomies (e.g., [9]) have been proposed, as discussed in [10].

Uncertainty gains more and more attention in recent years in both system and software engineering, especially for CPSs, which are required to be more and more context aware [62-64]. Moreover, CPSs inherently involve tight interactions between various engineering disciplines, information technology, and computer science. This magnifies uncertainties. Therefore, adequate treatment of uncertainty becomes increasingly more relevant for any non-trivial CPS. However, to the best of our knowledge, there is no comprehensive taxonomy of uncertainty existing in literature that focused specifically on CPS design or on system/software engineering in general. In this paper, we present such a taxonomy following a comprehensive literature review in various domains and investigating a number of industrial case studies. In the remainder of the section, we discuss how the concepts uncovered during the literature review align with our proposed taxonomy.

The concepts *BeliefAgent*, *BeliefStatement*, and *Belief* of the *Belief* part of our taxonomy are adapted from [12]. The author of [12] postulates that uncertainty involves a statement whose truth is expected by a person, and therefore the truth might differ for different persons (defined as *BeliefAgent* in our model). However, as we discussed in Section 3.1.2, we assigned a broader meaning to *BeliefAgent*: which can be an individual, a community of individuals, or a technology. The *U-Taxonomy* concepts *Environment* and *Locality* were adapted from [12, 65-67], and we related them to the other *U-Taxonomy* concepts.

Our knowledge taxonomy aligns well with the taxonomy of knowledge reported in [68]. Here the authors looked at how to manage different types of known and unknown knowledge to distinguish what is known from what is not known. *Knowledge* is also classified from a different perspective: something that everyone knows, tacit knowledge, conscious ignorance and meta-ignorance. Their objective is to better understand ignorance. The author of [69] also studied unknowns and provided a taxonomy particularly focusing on ignorance (named as *KnownUnknown* and *UnknownUnknown* in our taxonomy). In our taxonomy, we further elaborate these concepts and captured them as *KnowledgeType*, which is associated to *Evidence* and *IndeterminacySource* via *EvidenceKnowledge* and *IndeterminacyKnowledge*.

We classified uncertainties into various types including *Content*, *Time* and *Occurrence*. In [12], a chapter was dedicated to the discussion of data uncertainty and its measurement using *Measure*. The other two types of uncertainties were mentioned in [12, 15, 16], with examples but with no clear definitions provided. We adopted the measurement of *Measure* in our taxonomy but significantly extended it with *Function* (further classified as *Distribution* and *Entropy*) in addition to *Value*, to account for more complicated measurement of content, time and occurrence uncertainties.

Different types of sources of uncertainty for various purposes have been identified in the literature. In [70], the authors captured sources of uncertainty by considering risk and reliability analyses, based on which they classified uncertainty. The authors of [16, 71] identified sources of uncertainty in active systems. In [63, 72], the authors described the sources of uncertainty in software engineering in general. We however proposed the *U-Taxonomy* concepts *IndeterminacySource* and *IndeterminacyNature* to capture sources of uncertainty.

The author of [69] studied unknowns and provided a taxonomy particularly focusing on ignorance (named as *KnownUnknown* and *UnknownUnknown* in our taxonomy). *Ignorance* was classified as *Error* and *Irrelevance*. Error denotes incomplete knowledge, which was further classified as *Distortion* and

Incompleteness. *Incompleteness* refers to uncertainty and absence of data in their taxonomy. We aligned the measurement of uncertainty part of our taxonomy with the taxonomy of uncertainty in this work. That source also provides further classifications on *Irrelevance* and *Distortion*, which are irrelevant to our topic, and were consequently left out in our taxonomy. In [73], the author noted that uncertainty can occur in a random or systematic manner. In the *Pattern* part of our taxonomy, we further elaborated the "systematic" concept by introducing *Pattern*, and its sub categories.

In literature, uncertainty is often related to *Risk*. The acquisition project team of the US Air Force Electronic System Center (ESC) has proposed a risk matrix for evaluating risks [19]. They introduced the concepts of *Risk*, Impact, Likelihood of *Occurrence*, and Rate *of Risk* and also identified their relations. In our taxonomy, we reused these concepts and associated them with *Uncertainty*.

7 Conclusion

Cyber-Physical Systems (CPS) often consist of heterogeneous physical units (e.g., sensors, control modules) communicating via various networking equipment, interacting with applications and humans. Thus, uncertainty is inherent in CPSs due to tight interactions between hardware, software and humans, and being increasingly context aware. To this end, we presented a unified and comprehensive uncertainty taxonomy that we developed in the U-Test project, based on a thorough literature review of existing taxonomies from various domains (e.g., philosophy and healthcare).

References

[1] Broy, M. *Engineering Cyber-Physical Systems: Challenges and Foundations*. in *Proceedings of the Third International Conference on Complex Systems Design & Management CSD&M 2012*. 2013.

[2] Huang, H.-M., et al. *Cyber-Physical Systems for Real-Time Hybrid Structural Testing: A Case Study*. in *Proceedings of the 1st ACM/IEEE International Conference on Cyber-Physical Systems*. 2010.

[3] Tidwell, T., et al. *Towards Configurable Real-Time Hybrid Structural Testing: A Cyber Physical Systems Approach*. in *ISORC '09 Proceedings of the 2009 IEEE International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing*. 2009.

[4] Evans, P.C. and M. Annunziata, *Pushing the Boundaries of Minds and Machines*, in *General Electric (GE)*. 2012.

[5] Utting, M. and B. Legeard, *Practical Model-Based Testing: A Tools Approach*. 2006: Morgan-Kaufmann. 456.

[6] Chow, T.S., *Testing Software Design Modeled by Finite-State Machines.* IEEE Transactions on Software Engineering, 1978. 4(3): p. 178-187.

[7] Tannert, C., H.D. Elvers, and B. Jandrig, *The ethics of uncertainty.* EMBO reports, 2007. 8(10): p. 892-896.

[8] Mishel, M.H., *Uncertainty in illness.* Image: The Journal of Nursing Scholarship, 1988. 20(4): p. 225-232.

[9] Babrow, A.S., C.R. Kasch, and L.A. Ford, *The many meanings of uncertainty in illness: Toward a systematic accounting.* Health communication, 1998. 10(1): p. 1-23.

[10] Han, P.K., W.M. Klein, and N.K. Arora, *Varieties of Uncertainty in Health Care A Conceptual Taxonomy.* Medical Decision Making, 2011. 31(6): p. 828-838.

[11] Ali, S., L.C. Briand, and H. Hemmati, *Modeling robustness behavior using aspect-oriented modeling to support robustness testing of industrial systems.* Software & Systems Modeling, 2012. 11(4): p. 633-670.

[12] Lindley, D.V., *Understanding uncertainty (revised edition)*. 2014: John Wiley & Sons.

[13] Bammer, G. and M. Smithson, *Uncertainty and risk: multidisciplinary perspectives*. 2012: Routledge.

[14] Potter, K., P. Rosen, and C.R. Johnson, *From quantification to visualization: A taxonomy of uncertainty visualization approaches*, in *Uncertainty Quantification in Scientific Computing*. 2012, Springer. p. 226-249.

[15] Taylor, B.N., *Guidelines for Evaluating and Expressing the Uncertainty of NIST Measurement Results (rev*. 2009: DIANE Publishing.

[16] Wasserkrug, S., A. Gal, and O. Etzion, *A taxonomy and representation of sources of uncertainty in active systems*, in *Next Generation Information Technologies and Systems*. 2006, Springer. p. 174-185.

[17] Cimatti, A., A. Micheli, and M. Roveri. *Timelines with Temporal Uncertainty*. in *Aaai*. 2013.

[18] Sprunt, B., L. Sha, and J. Lehoczky, *Scheduling sporadic and aperiodic events in a hard real-time system*. 1989, DTIC Document.

[19] Garvey, P.R. and Z.F. Lansdowne, *Risk matrix: an approach for identifying, assessing, and ranking program risks.* Air Force Journal of Logistics, 1998. 22(1): p. 18-21.

[20] Dumbravă, V. and V.-S. Iacob, *Using Probability–Impact Matrix in Analysis and Risk Assessment Projects.* Journal of Knowledge Management, Economics and Information Technology, 2013. 3(6).

[21] Fu, S., H. Zhou, and Y. Xiao. *The application of a risk matrix method on campus network system risk assessment*. in *Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on*. 2011. IEEE.

[22] Amland, S., *Risk-based testing:: Risk analysis fundamentals and metrics for software testing including a financial application case study.* Journal of Systems and Software, 2000. 53(3): p. 287-295.
[23] Klir, G., *Facets of systems science*. Vol. 7. 2013: Springer Science & Business Media.
[24] Avizienis, A., et al., *Basic concepts and taxonomy of dependable and secure computing.* Dependable and Secure Computing, IEEE Transactions on, 2004. 1(1): p. 11-33.
[25] Eugster, P.T., et al., *The many faces of publish/subscribe.* ACM Computing Surveys (CSUR), 2003. 35(2): p. 114-131.
[26] Magazin, M.M.N. *m2m now*. March 2015]; Available from: http://www.m2mnow.biz/2014/08/05/23417-employee-safety-security-regulations-raise-stakes-fleet-operators/.
[27] Aslam, T., I. Krsul, and E.H. Spafford, *Use of a taxonomy of security faults.* 1996.
[28] łgorzata Steinder, M. and A.S. Sethi, *A survey of fault localization techniques in computer networks.* Science of computer programming, 2004. 53(2): p. 165-194.
[29] Leszak, M., D.E. Perry, and D. Stoll. *A case study in root cause defect analysis*. in *Proceedings of the 22nd international conference on Software engineering*. 2000. ACM.
[30] Avizienis, A., J.-C. Laprie, and B. Randell, *Fundamental concepts of dependability*. 2001: University of Newcastle upon Tyne, Computing Science.
[31] Strong, D.M., Y.W. Lee, and R.Y. Wang, *Data quality in context.* Communications of the ACM, 1997. 40(5): p. 103-110.
[32] Buchholz, T., A. Küpper, and M. Schiffers. *Quality of context: What it is and why we need it*. in *Proceedings of the workshop of the HP OpenView University Association*. 2003.
[33] Weber, R.H., *Internet of things–Governance quo vadis?* Computer Law & Security Review, 2013. 29(4): p. 341-347.
[34] Plato, S. *Stanford Plato Phenomenolgy*. Available from: http://plato.stanford.edu/entries/phenomenology/.
[35] Gray, J. *Why do computers stop and what can be done about it?* in *Symposium on reliability in distributed software and database systems*. 1986. Los Angeles, CA, USA
[36] Atzori, L., A. Iera, and G. Morabito, *The internet of things: A survey.* Computer networks, 2010. 54(15): p. 2787-2805.
[37] Yuriyama, M. and T. Kushida. *Sensor-cloud infrastructure-physical sensor management with virtualized sensors on cloud computing*. in *Network-Based Information Systems (NBiS), 2010 13th International Conference on*. 2010. IEEE.
[38] Stuedi, P., I. Mohomed, and D. Terry. *WhereStore: Location-based data storage for mobile devices interacting with the cloud*. in *Proceedings of the 1st ACM Workshop on Mobile Cloud Computing & Services: Social Networks and Beyond*. 2010. ACM.
[39] Satyanarayanan, M., et al., *The case for vm-based cloudlets in mobile computing.* Pervasive Computing, IEEE, 2009. 8(4): p. 14-23.
[40] Cuervo, E., et al. *MAUI: making smartphones last longer with code offload*. in *Proceedings of the 8th international conference on Mobile systems, applications, and services*. 2010. ACM.
[41] Bhardwaj, K., et al. *ECC: Edge Cloud Composites*. 2014.
[42] Nastic, S., et al. *Provisioning Software-defined IoT Cloud Systems*. 2014. IEEE.
[43] Bonomi, F., et al. *Fog computing and its role in the internet of things*. in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*. 2012.
[44] Nastic, S., et al. *rtGovOps: A Runtime Framework for Governance in Large-scale Software-defined IoT Cloud Systems*.
[45] Cheng, B.C., et al., *A Goal-Based Modeling Approach to Develop Requirements of an Adaptive System with Environmental Uncertainty*, in *Model Driven Engineering Languages and Systems*, A. Schürr and B. Selic, Editors. 2009, Springer Berlin Heidelberg. p. 468-483.

[46] Cheng, B.H.C., *Tackling Uncertainty for Transportation Cyber-Physical Systems*, in *National Workshop on Transportation CyberPhysical Systems*. 2014.
[47] Technology, N.I.o.S.a., *Foundations for Innovations in Cyber-Physical Systems*. 2013.
[48] Ramirez, A.J., et al. *Automatically exploring how uncertainty impacts behavior of dynamically adaptive systems*. in *Automated Software Engineering (ASE), 2011 26th IEEE/ACM International Conference on*. 2011.
[49] Whittle, J., et al. *RELAX: Incorporating Uncertainty into the Specification of Self-Adaptive Systems*. in *Requirements Engineering Conference, 2009. RE '09. 17th IEEE International*. 2009.
[50] Rajkumar, R., et al. *Cyber-physical systems: The next computing revolution*. in *Design Automation Conference (DAC), 2010 47th ACM/IEEE*. 2010.
[51] Sztipanovits, H.G.a.B.R.a.B.S.a.J., *Science and Engineering of Cyber-Physical Systems (Dagstuhl Seminar 11441)*, in *Dagstuhl Reports*, H.G.a.B.R.a.B.S.a.J. Sztipanovits, Editor. 2012, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik: Dagstuhl, Germany. p. 22.
[52] Kremer, U., *Cyber-Physical Systems: A case for soft real-time*.
[53] Neuman, C. *Challenges in security for cyber-physical systems*. in *DHS: S&T workshop on future directions in cyber-physical systems security*. 2009. Citeseer.
[54] Pinto, A. and S. Krishnamurthy. *Developing design tools for uncertain systems in an industrial setting*. in *Communication, Control, and Computing (Allerton), 2010 48th Annual Allerton Conference on*. 2010.
[55] Ramirez, A.J., A.C. Jensen, and B.H. Cheng. *A taxonomy of uncertainty for dynamically adaptive systems*. in *Software Engineering for Adaptive and Self-Managing Systems (SEAMS), 2012 ICSE Workshop on*. 2012. IEEE.
[56] Lui, S., et al. *Cyber-Physical Systems: A New Frontier*. in *Sensor Networks, Ubiquitous and Trustworthy Computing, 2008. SUTC '08. IEEE International Conference on*. 2008.
[57] Talcott, C., *Cyber-Physical Systems and Events*, in *Software-Intensive Systems and New Computing Paradigms*, M. Wirsing, et al., Editors. 2008, Springer Berlin Heidelberg. p. 101-115.
[58] Walker, W.E., et al., *Defining uncertainty: a conceptual basis for uncertainty management in model-based decision support.* Integrated assessment, 2003. 4(1): p. 5-17.
[59] Brown, J.D., *Knowledge, uncertainty and physical geography: towards the development of methodologies for questioning belief.* Transactions of the Institute of British Geographers, 2004. 29(3): p. 367-381.
[60] Perez-Palacin, D. and R. 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*. 2014, ACM: Dublin, Ireland. p. 3-14.
[61] IEC/ISO, *IEC 31010:2009 - Risk management -- Risk assessment techniques*. 2009.
[62] Rajkumar, R.R., et al. *Cyber-physical systems: the next computing revolution*. in *Proceedings of the 47th Design Automation Conference*. 2010. ACM.
[63] Conti, M., et al., *Looking ahead in pervasive computing: Challenges and opportunities in the era of cyber–physical convergence.* Pervasive and Mobile Computing, 2012. 8(1): p. 2-21.
[64] Garlan, D. *Software engineering in an uncertain world*. in *Proceedings of the FSE/SDP workshop on Future of software engineering research*. 2010. ACM.
[65] Hu, F., *Cyber-Physical Systems: Integrated Computing and Engineering Design*. 2013: CRC Press.
[66] Cheng, B.H.C., et al., *A goal-based modeling approach to develop requirements of an adaptive system with environmental uncertainty*, in *Model Driven Engineering Languages and Systems*. 2009, Springer. p. 468-483 %@ 3642044247.
[67] Wan, K., K.L. Man, and D. Hughes, *Specification, analyzing challenges and approaches for cyber-physical systems (CPS).* Engineering Letters, 2010. 18(3): p. 308 %@ 1816-093X.

[68] Kerwin, A., *None Too Solid Medical Ignorance.* Science Communication, 1993. 15(2): p. 166-185.
[69] Smithson, M., *Ignorance and uncertainty: Emerging paradigms*. 1989: Springer-Verlag Publishing.
[70] Der Kiureghian, A. and O. Ditlevsen, *Aleatory or epistemic? Does it matter?* Structural Safety, 2009. 31(2): p. 105-112 %@ 0167-4730.
[71] de Lemos, R., et al. *Software engineering for self-adaptive systems*. 2009. Springer.
[72] Ziv, H., D. Richardson, and R. Klösch. *The uncertainty principle in software engineering*. 1997.
[73] Bell, S., *A beginner's guide to uncertainty of measurement*. 2001: National Physical Laboratory Teddington, Middlesex.