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

D 2.2 - Report on Uncertainty Modelling Framework V.2

Project Acronym	U-Test	**Grant Agreement Number**		H2020-ICT-2014-1. 645463	
Document Version	0.5	**Date**	2016-10-28	**Deliverable No.**	2.2
Contact Person	Hong-Linh Truong	**Organisation**		TUW	
Phone	+43-1-58801-18456	**E-Mail**		truong@dsg.tuwien.ac.at	

Document Version History

Version No.	Date	Change	Author(s)
0.1	2016-07-01	Initial document outline	Georgiana Copil
0.2	2016-10-10	Integration of contributions from partner	All
0.3		Internal review version	All
0.4	2016-10-25	Addressed internal review comments. Request for new inputs	Hong-Linh Truong
0.5	2016-10-28	Address all comments- Final version	All

Contributors

Name	Partner	Part Affected	Date
Georgiana Copil	TUW	All	
Hong-Linh Truong	TUW	All	
Ivan Pavkovic	TUW	All	
Marc-Florian Wendland	Fraunhofer FOKUS	Sections 3.2.1.2, 3.2.3.1, 4.1.1, 4.2.1, 5.1	
Shaukat Ali	SRL	Sections 3.2.3,4.1.3,4.1.4, 4.2.3,4.1.4,5.3	
Man Zhang	SRL	Sections 3.2.3,4.1.3,4.1.4, 4.2.3,4.1.4,5.3	
Tao Yue	SRL	Sections 3.2.3,4.1.3,4.1.4, 4.2.3,4.1.4,5.3	
Phu Hong Nguyen	SRL	Sections 3.2.3,4.1.3,4.1.4, 4.2.3,4.1.4,5.3	

Table of Contents

1 Executive Summary

This deliverable describes the second version of the U-Test Uncertainty Modelling Framework (UMF V.2). This U-Test UMF V.2 provides a specification of uncertainty concepts as Unified Modelling Language (UML) profile. In addition to the profile a modelling methodology has been defined in order to guide test engineers through the UMF components. Furthermore, this deliverable defines the test-ready models of the U-Test pilot cases.

In addition to the profile and test ready models included in UMF V.1, UMF V.2 provides refined profiles, and a new set of test-ready models, achieving milestone M3: (i) at application level, 50% and 66% of the ULMA and respectively FPX use cases, (ii) at integration level, 66% of the ULMA and FPX use cases, and (iii) at infrastructure level, 60% of the ULMA and FPX use cases.

2 Introduction

This document presents the effort achieved from D2.1 [1], in M18-M22. The Uncertainty Modeling Framework (UMF) provides an approach to create and specify test-ready models based on existing modeling and testing standards. The models are based on the U-test specific uncertainty profile providing the relevant concepts to describe uncertainty at Application level, Infrastructure level, and Integration level. This report presents the second version of the UMF, achieved through iterative improvements over D2.1. The final version of the UMF will be presented in D2.3.

2.1 U-Test Workflow

Figure 1 shows the general U-Test workflow, the place of the Uncertainty Modeling Framework (UMF), and its relationship with other U-Test components.

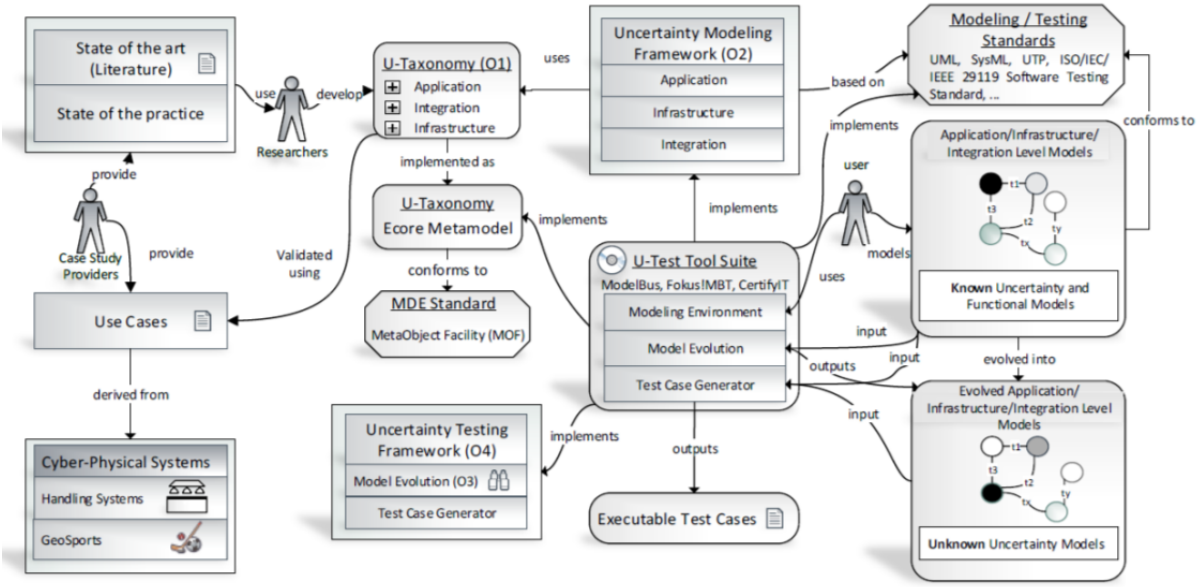

Figure 1 U-Test workflow

The UMF is using the U-Taxonomy [6] defined through analyzing the state of the art and state of the practice, including a classification of concepts related to uncertainty in cyber-physical systems (CPS). Based on concepts introduced in U-Taxonomy and Modeling and Testing Standards, the UMF comprises models and profiles for uncertainty in CPS. The output of the UMF are test-ready models based on this uncertainty profile. The test-ready models describe the Geo Sports and Warehouse Management System scenarios, and the use cases described for them [5].

The models resulting from using the UMF are the input to the Uncertainty Testing Framework UTF (WP4). The UTF offers corresponding uncertainty test case generators to generate and execute adequate test cases for the U-Test use cases. Finally, the test execution results are used to evolve unknown uncertainty information (before the U-Test workflow) into known uncertainty functional models (after a walk-through to the U-Test workflow). This means that formerly unknown uncertainty behavior went into known (uncertainty) behavior.

2.2 Objectives of the Deliverable

This report is a deliverable of work package "Developing Modeling Methodologies for Uncertainty Testing" (WP2) of the U-Test project. The overall objective is to provide a systematic way of standard-based holistic modelling of uncertainties and uncertain behaviors in CPS at three levels namely application (task 2.1), infrastructure (task 2.2) and integration (task 2.3). Therefore, this deliverable reports the results of the modelling support (uncertainty profile and methodology) developed in all levels. Moreover, this deliverable provides a description of the defined test-ready models for several use cases of the two U-Test pilots/scenarios at the Application, Infrastructure, and Integration levels.

2.3 Structure of the Deliverable

The rest of this deliverable is structured as follows: Section 3 presents the Uncertainty Modeling Framework (UMF), focusing on the updates from D2.1 [1], Section 4 describes the progress achieved with Pilot Modeling, and Section 5 concludes this document, summarizing the achievements and the future work.

3 Uncertainty Modeling Framework

In this section, we will present the current status of the U-test Uncertainty Modeling Framework (UMF) and its updates.

3.1 Overview

Figure 2 shows the architectural overview of the UMF, together with its inputs and outputs. This deliverable uses as input updated requirements, together with the uncertainty taxonomy. The requirements for both the Geo Sports and Warehouse Management System scenarios have been updated with supplementary details. Furthermore, in the Geo Sports scenario, for supporting the test execution, in some of the use-cases the X4 device was replaced with Quuppa [1]. These changes are due to use case requirements. This resulted of course in supplementary updates of the requirements detailing the behavior of the Quuppa device. The UML is built on well-established and widely accepted modeling standards such as UML, UTP, and MARTE.

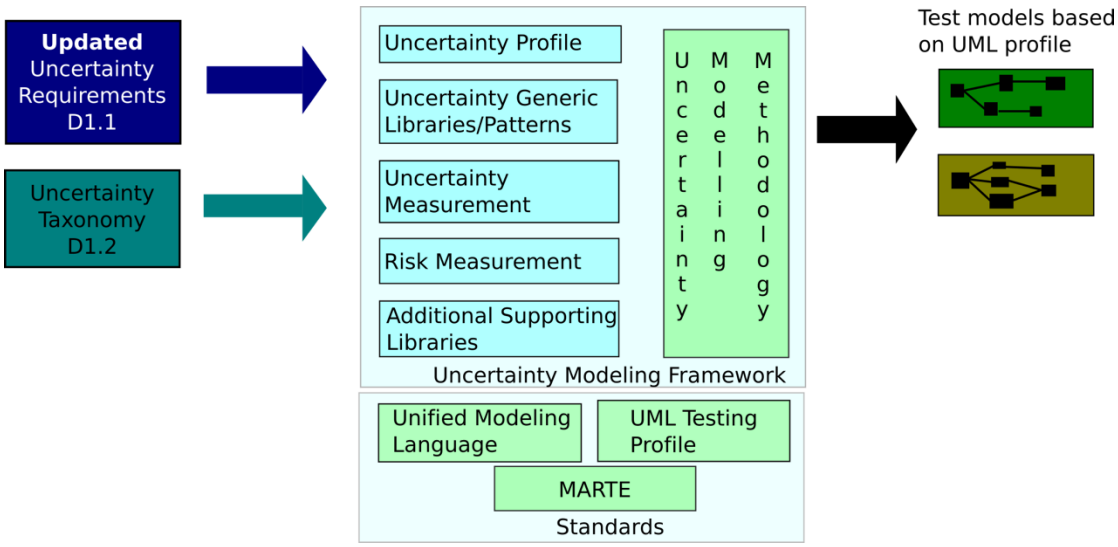

Figure 2: UMF Architecture

Our UMF defines a UML profile that provides support for uncertainty definition, management and specification. The UMF also provides methodologies for easing the usage of and application of uncertainty profiles. Based on the UMF we develop test-ready models, for both pilot scenarios, which will be described in Section 4.

3.2 UMF components

In this section we will describe the updates to the UML Uncertainty Profile, Model Libraries and Modelling Methodology with respect to D2.1 [1].

3.2.1 UML Uncertainty Profile

The U-Test UML Uncertainty Profile includes (1) Core Profile, (2) Application Level Profile, and (3) Infrastructure Level Profile.

3.2.1.1 Core Profile

This section only highlights the updates as compared to D2.1 to avoid unnecessary repetition. More specifically, there are some minor updates in the core profile for modeling Belief and Uncertainty.

During creating test-ready models, we discovered that it is reasonable to define Indeterminacy Source as a constraint, which can be used, for example, to generate test data to actually introduce indeterminacy during test execution. <<The details have been removed from this document due to the agreement for confidentiality with the pilot companies>>. According to the experience of case studies, we modified UML profile diagram for Belief Modeling as shown in Figure 3. In this diagram, we

A-1. introduced a new extension from the *Constraint* metaclass to «IndeterminacySource»

A-2. added new associations from *Uncertainty* to *Constraint* as "referredIndetermiancySourceInConstraint" attribute.

A-3. changed the role name of "referredIndetermiancySource" to "referredIndetermiancySourceInClassifier" in the association from *Uncertainty* to *Classifier*.

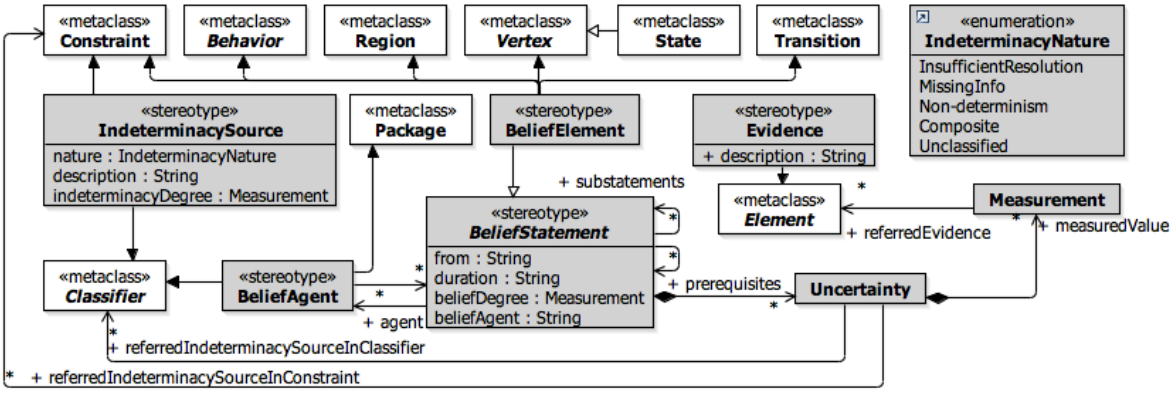

Figure 3. UML profile diagram for Belief modeling

On the other hand, five redundant extensions of UUP at the Integration level are deleted since those are already covered by extension to the Element metaclass. These are summarized below:

D-1. Deleted the *uml::Package* as the Extension for *«Cause»*

D-2. Deleted the *uml::Package* as the Extension for «Effect»

D-3. Deleted the *uml::Package* as the Extension for «Evidence»

D-4. Deleted the *uml::Package* as the Extension for «Lifetime»

D-5. Deleted the *uml::Package* as the Extension for «Pattern»

For a complete and updated specification of the Integration (core) level UMF, readers can find all the details in our first technical report for this deliverable [3]. This technical report includes the updated UML Uncertainty Profile (UUP) as well as the modeling methodology.

In addition, we documented a second technical report that describes the additional details on: 1) Explanation of methodology and profile applications with examples, 2) Some preliminary evaluation of the modeling methodology. This second technical report is under submission to a journal and can be found in [4].

3.2.1.2 Application Level Profile

The Application level part of the UML Uncertainty Profile (UUP) has been entirely redesigned compared to D2.1. The reasons for the complete re-design were due to applicability issues that we encountered when working on the Application level use cases. On the other hand, information required for the exact description of uncertainties and processing thereof in the context of test case generation was just missing in the D2.1.

The UUP application level profile consists of three parts:

- Application level uncertainties;
- Fitness factor calculation; and
- Execution invariants.

These parts are briefly described in the next sections.

3.2.1.2.1 Application level uncertainties

The Application level uncertainties concepts are used for modelling application level uncertainties and all related aspects we require for uncertainty testing generation[1]. The abstract syntax of the UUP application level is shown in Figure 4.

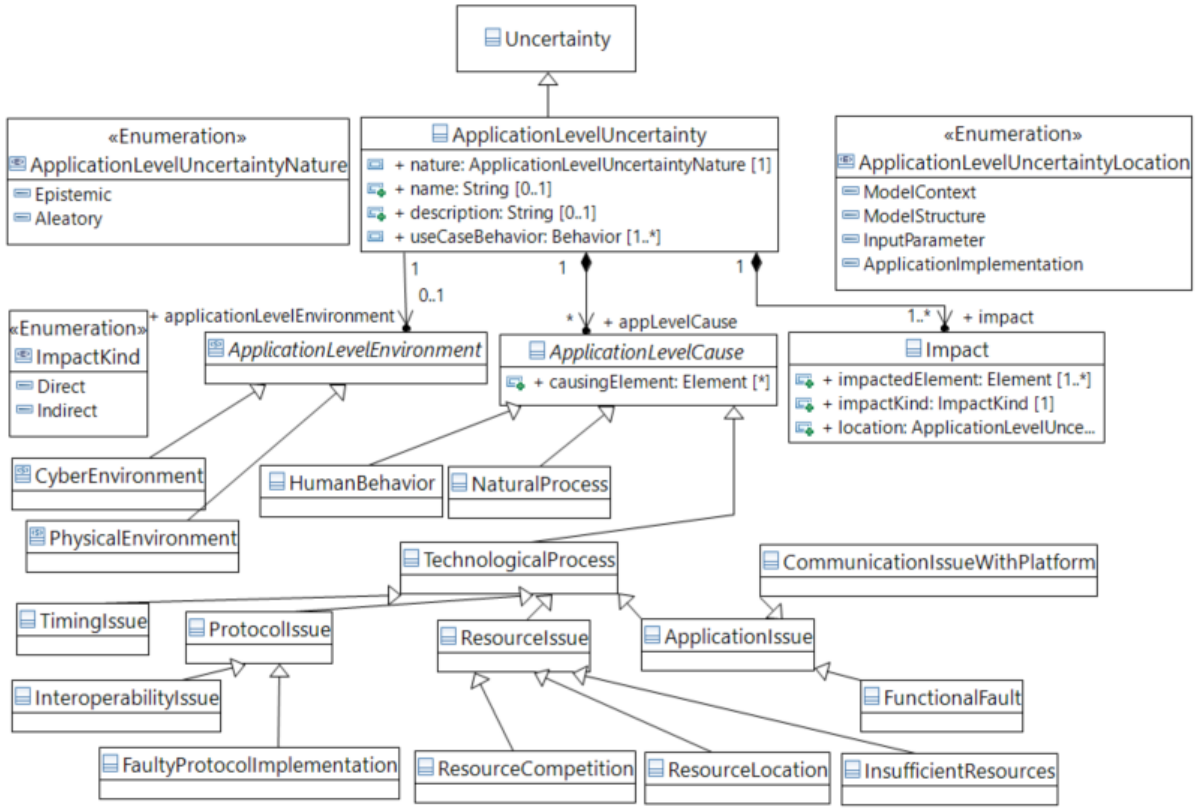

Figure 4. UML Uncertainty Profile (UUP) for the application Level – Application level uncertainties

The metaclass *ApplicationLevelUncertainty* has been changed to a subclass of *core::Uncertainty* instead of composing instances thereof. By doing so, the treatment (i.e., creation and handling) of uncertainties are consistent the different levels. The attributes *name* and *description* have been introduced in order to make uncertainties better readable

[1] Uncertainty testing comprises modelling of uncertainties and believes/believe statements, creation of test models containing these uncertainties, test case generation, test case execution and test evaluation eventually.

and understandable by uncertainty testers. An uncertainty is supposed to occur or actually occurred in one or more use case scenarios. In order to relate uncertainties to those behavioral descriptions, the property *useCaseBehavior* has been introduced.

The enumeration *ApplicationLevelUncertainty* remains unchanged. The enumeration *ApplicationLevelUncertaintyLocation* has been newly introduced, but stems from a formerly representation as classes.

The abstract metaclass *ApplicationLevelEnvironment* and its concrete subclasses *CyberEnvironment* and *PhysicalEnvironment* remain unchanged.

The metaclass Impact has been changed from D2.1 where it has been a subclass of *core::Impact*. This metaclass, however, was removed from the core part. Similar to D2.1, the metaclass *Impact* defines the kind of the impact (direct or indirect as indicated in the enumeration *ImpacttKind*) as well as the elements that are impacted by the uncertainty.

The abstract metaclass *ApplicationLevelCause* has been slightly modified with respect to our needs. An application level cause is now able to point to one or more elements that are supposed to cause the uncertain behaviors, which are then impacting the impacted elements identified by the metaclass *Impact*. In D2.1, this reference was a simple string, hence, inappropriate for a pure model-based engineering approach as we follow it. Finally, several further technological process causes have been added to the profile. These causes stem first and foremost from the experiences we made with the UTEST pilot use cases.

3.2.1.2.2 Fitness factor facility

A complete newly developed and formalized part of the UUP application level is the UUP *fitness factor facility*. Its abstract syntax is depicted in Figure 5. It provides the uncertainty tester with the abilities to define the functions that are eventually called from the test automation architecture in order to calculate the fitness factor. This part is essential for our methodology of search-based testing.

A *MetricGoal* is an operation that calculates a fitness value in order to decide whether the value is above or below the specified threshold. The values used by a metric function are obtained from so called *FitnessFactorProviders*. Fitness factor values are described by both the expected response of the test item as defined in the test case and the actual response of the test item as captured in the test execution logs.

Metric function operations are later on invoked by arbitration specifications in order to determine whether currently unknown behaviors have been discovered during test case execution. Based on such a statement, the evolutionary test case generation is steered.

A more detailed description of the methodology behind the UUP application level fitness factor facility can be found in Section 4.2.3.5. Fitness of UTEST D 3.1 - Report on Uncertainty Testing Framework V.1.

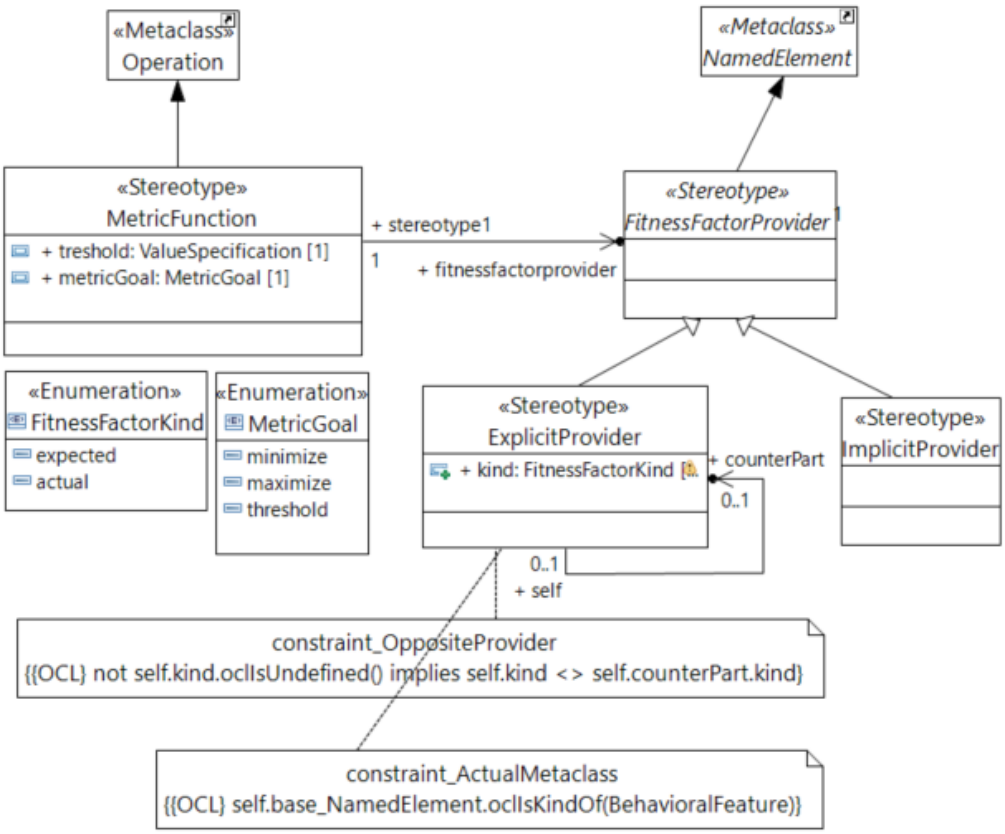

Figure 5. UML Uncertainty Profile (UUP) for application level – Fitness factor facility

3.2.1.2.3 Execution invariants

The concepts to describe execution invariants were newly developed after D2.1. Its (very concise) abstract syntax is shown in Figure 6.

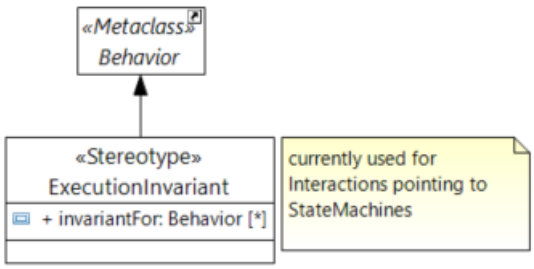

Figure 6. UML Uncertainty Profile (UUP) for application level - Execution invariants

3.2.1.3 Infrastructure Level Profile

Figure 7 shows the profile for concepts necessary for expressing uncertainty-related concepts with regards to the infrastructure. The uncertainty at the infrastructure level affects cyber-physical systems both functionally and in their non-functional behaviors. Furthermore, they can produce effects only inside the CPS, or result in failures outside of the CPS (IngressEgress type). The locality of the uncertainty is very important at the infrastructure level. The profile has no significant changes compared to the version introduced in D2.1 [1].

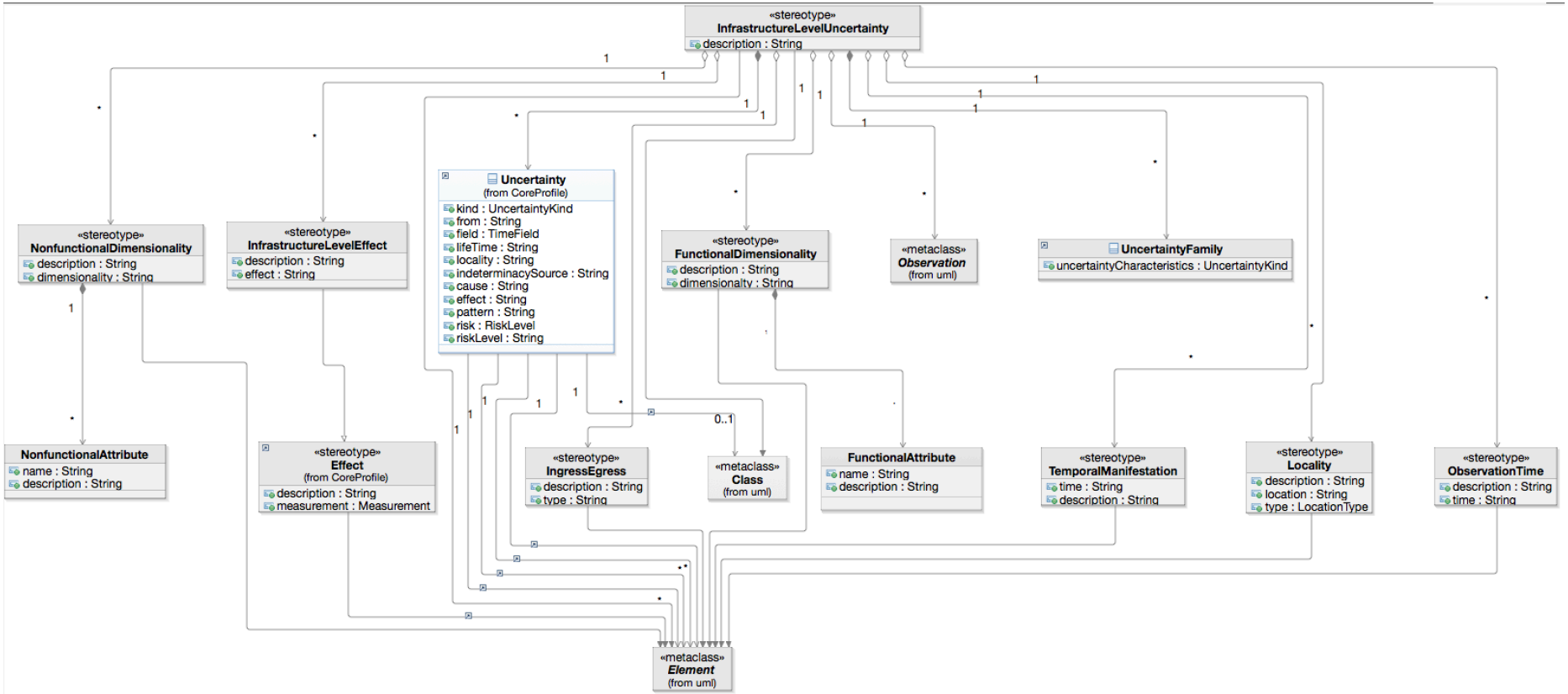

Figure 7: Infrastructure level uncertainty

3.2.2 Cyber-physical System Profile

The CPS profile shown in Figure 8 includes the most important concepts and stereotypes necessary for modeling cyber-physical systems (CPS). The initial version of the CPS profile was presented in D2.1 [1]. In this document we focus on the updates from the initially presented version.

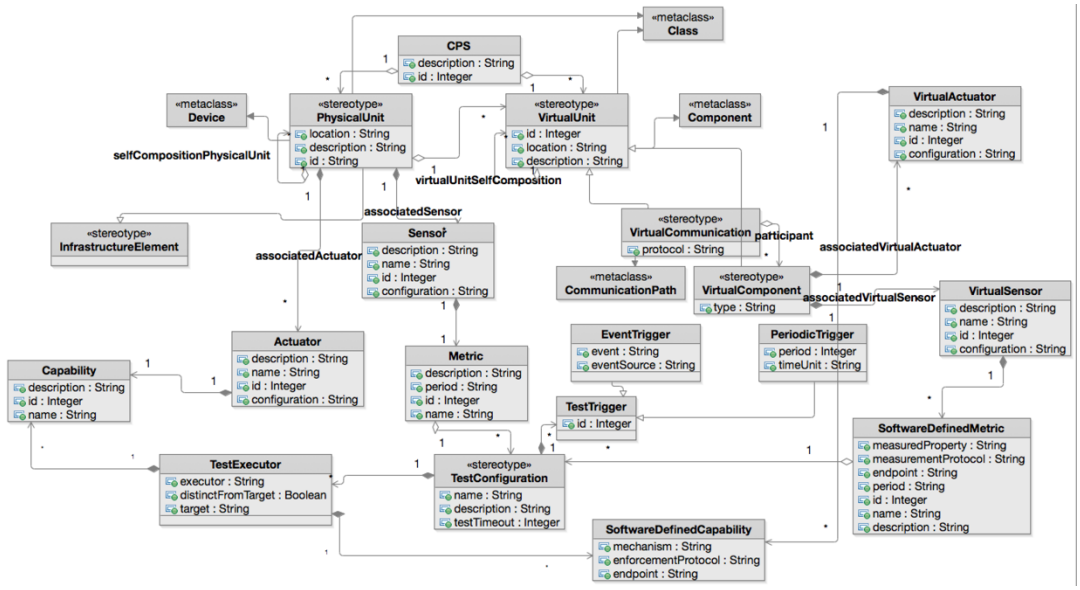

Figure 8: Cyber-physical Profile

The refinements to the CPS profile include adding supplementary attributes for modeling various concepts, such as id for physical and virtual units, name and id for metrics and software defined metrics.

Furthermore, we have included some concepts that are necessary for generating tests associated to the cyber-physical systems. The tests will reference metrics, capabilities and configurations of units that construct the cyber-physical system.

The TestExecutor describes how the generated test can be executed. It has as attributes an executor, depicting who is executing the test, target depicting the device/CPS part/unit that needs to be tested, distinctFromTarget depicting whether the executor is different from the target of the test.

The TestConfiguration gives the configuration details necessary for running the test associated with the CPS: its name, description, and the testTimeout which gives the amount of time one should wait for the respective test, until considering it as un-responding and failed.

The TestConfiguration is composed of one or more TestTriggers, which can be of two types: EventTrigger and PeriodicTrigger. The EventTrigger describes the event as a response to which the test should be executed. It has associated the event description and the eventSource. The PeriodicTrigger is associated with tests that need to be executed at fixed time intervals. Therefore, the PeriodicTrigger describes the period and the timeUnit at which the test should be repeated.

3.2.3 Modelling Methodology

In addition to the uncertainty profile definition, the UMF provides also a modelling methodology to guide test modeler through the U-Test workflow. The modeling methodology is described in D2.1.

3.2.3.1 Updates of the Application level modelling methodology

The U-Test Application level modelling methodology has been significantly changed from what was reported in D2.1 due to experiences made with the pilots. The Application level methodology now follows a sequential process that consists of three main steps:

- Pilot modelling process;
- Uncertainty modelling process;
- Deployment modelling process.

These modelling sub-processes are described in greater detail in the next subsections.

3.2.3.1.1 Pilot modelling process

The U-Test Application level pilot modelling process deals with the creation of pilot models that formalize the use cases descriptions reported in D1.1 (and its revisions). The first step is to identify the system part that will assume the role of the test item in one or more use cases. We currently do not utilize communicating state machines, but model everything we need into a single state machine that represents the test item's expected input/output behavior. Since the models rather reflect the expected behavior from a tester's point of view, we call the resulting model a test model. After the system part was identified and modeled, a test configuration is created that owns a part of type of the system part. This part (compositional property) is stereotyped as <<test item>>.

Next step is to analyze the use case scenario descriptions and to formalize the basic scenario flow as UML state machine. The sole purpose of this state machine is to describe the expected input/output behavior of the test item. Therefore, the tester has to decide for each transition, whether the transition is triggered by an external entity (i.e., other system parts emulated later on by test components), by some internal processes, by a timed event or simply by a completion event. In the first case, the tester has to reason about the required interface operation that is offered and eventually supposed to be invoked by the system part's environment. Once the interface operation is clear, the tester has to reason about the types (and gradually the entire type system) that are going to be submitted as formal parameters. Once the interface operation is clear, the tester has to decide whether there exist some triggering conditions for that transition and has to reflect these as fUML-compliant Activity. Afterwards the transition's effect needs to be reflected in the model. We are using fUML-compliant activities to have a platform-independent representation of both the logical types and the exploration of the test item's state space.

Nested states, compound transitions and do/entry/exit behaviors are currently not supported.

3.2.3.1.2 Uncertainty modelling

Uncertainties belong to believe statements and relate to believe agents. In our methodology, a believe agent is modeled as package that contains a number of believe statements. A believe statement is reflected as Constraint with <<BelieveStatement>> applied. The believe statement then contains an instance of an application level uncertainty. The main idea of our approach is that the specific uncertainty flows of the use case descriptions are not modelled explicitly in the state machines, but inferred from a modelled uncertainty via state machine modification (see D3.1 for further details).

Afterwards the tester has to determine multiple information about the application level uncertainty such as the location where the uncertainty reveals, the impact and the impacting element, the cause and causing element etc.

3.2.3.1.3 Deployment modelling

The U-Test Application level methodology claims to be fully automated after the test model design activities have been finished. Therefore, it is necessary to determine the elements that shall be passed to the test automation framework. To stick with a pure model-based methodology, these elements are identified by means of model elements as well. For one, we are using dedicated U-Test directives (represented by the stereotype «UTestDirective» which integrates with the UTP test design facility) to select uncertainties and state machines, define coverage criteria and test design techniques. On the other, UML deployment concepts are used to eventually define the effective deployment location and artifacts. Both U-Test directives and deployment information are passed to the test automation framework finally that is responsible to perform automated test design, automated test implementation, automated test execution and automated test evaluation.

3.2.3.2 Updated in the Integration Level Methodology

3.2.3.2.1 Updates in the Lifetime/Cause/Pattern/Effect of Uncertainty Modeling

Compared to the modeling methodology reported in D2.1, we have made several updates aiming at better modeling the Lifetime/Cause/Pattern/Effect of Uncertainty and the IndeterminacySource.

Regarding the update for modeling the Lifetime/Cause/Pattern/Effect of Uncertainty, the option 3 is newly added, as shown in Figure 9. In the new option of modeling, first, the «Cause» is mandatory to be applied on the transition when it transits to the different target states (S1.3.1). The next step of option 3 is to link transition applied «Cause» to Uncertainty via “referredCause” attribute.

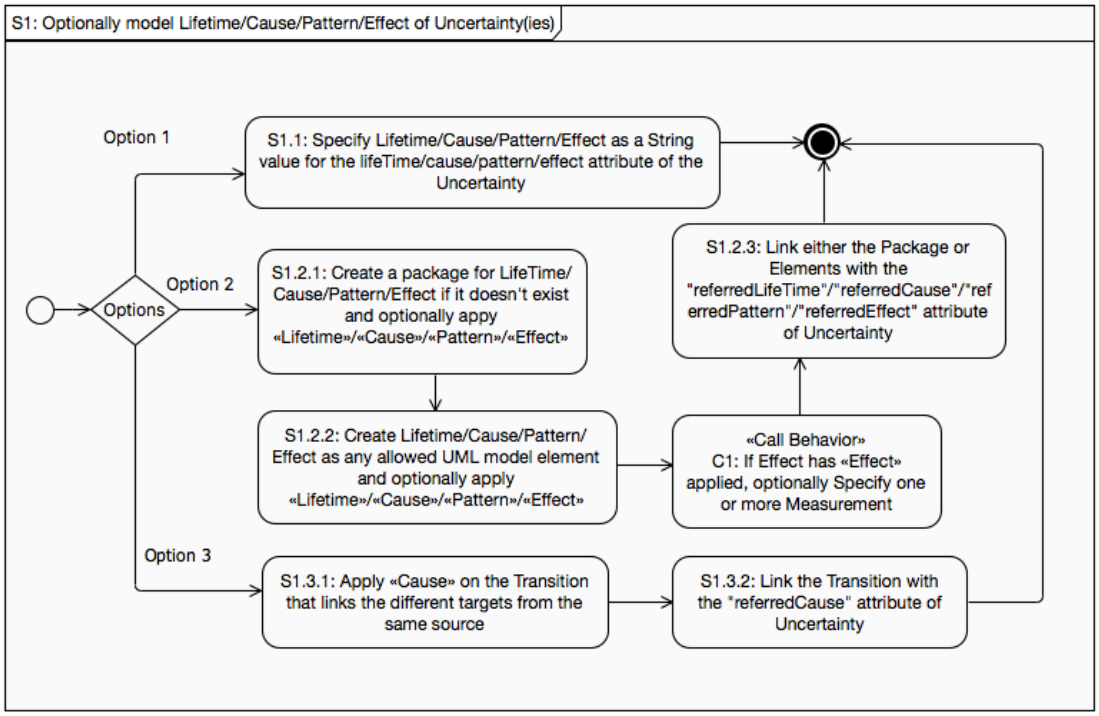

Figure 9. Modeling Lifetime/Cause/Pattern/Effect of Uncertainty

The update aims at better modeling IndeterminacySource is described in the following section.

3.2.3.2.2 Updates in the IndeterminacySource Modeling

A2.2.2.2 is newly added as shown in Figure 10. The new option allows the creation of a constraint to represent an indeterminacy source and apply «IndeterminacySource» to it (A2.2.2.2), specify the nature and description of each indeterminacy source (A2.2.3), specify measurements for each indeterminacy source (C1), and associate the created constraints to the "referredIndeterminacySourceInConstraint" attribute of Uncertainty.

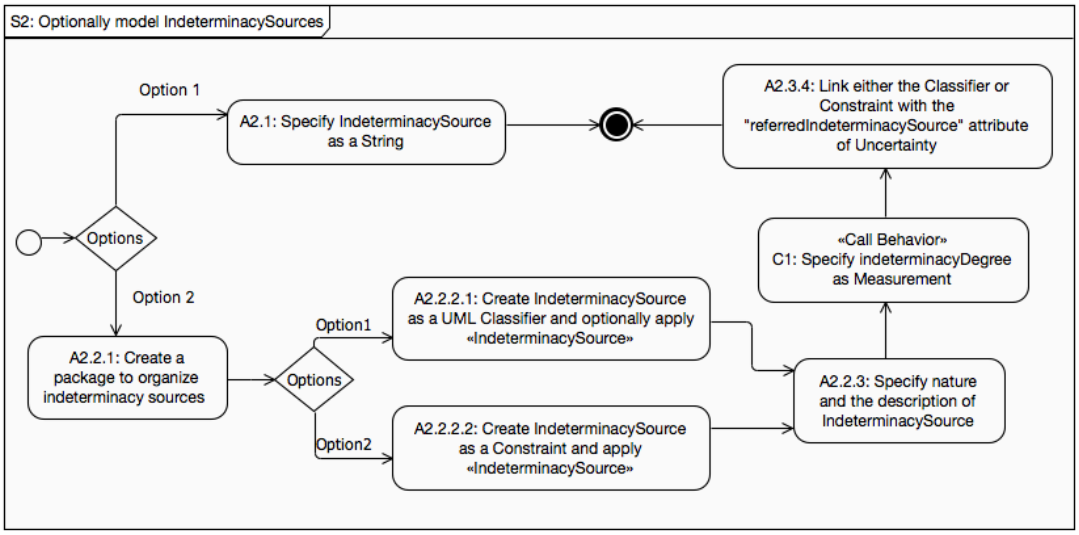

Figure 10. Modeling IndeterminacySource

Table 1 gives a summary of the updates in the UMF at the Integration level.

Table 1. Overall Updates in the UMF

Category	Name	Update Status	Details in Section #
Profile	«IndeterminacySource»	Updated	Section 3.2.1.1, point A-1
	Uncertainty	Updated	Section 3.2.1.1, point A-2, A-3
	«Cause»	Updated	Section 3.2.1.1, point D-1
	«Effect»	Updated	Section 3.2.1.1, point D-2
	«Evidence»	Updated	Section 3.2.1.1, point D-3
	«Lifetime»	Updated	Section 3.2.1.1, point D-4
	«Pattern»	Updated	Section 3.2.1.1, point D-5
Methodology	Cause Modeling	Updated	Section 3.2.3.2.1
	IndeterminacySource Modeling	Updated	Section 3.2.3.2.2

3.2.3.3 Update of the Infrastructure level modelling methodology

We do not have any update in our methodology, although profiles have been revised and improved.

3.2.4 Test-Ready Models for Pilots

The output of the UMF are test-ready models, which are based on UML, UTP and the UUP profile. In general, test modelers should be guided by the provided modeling methodology in order to create these test ready models. The goal is to produce models that are defined at a sufficient level of detail to generate adequate test cases. The test-ready models of the both pilots are described in Section 4.

4 Pilot Modelling

This chapter describes the defined test-ready models based on the U-Test workflow and the modelling methodology. In this deliverable, the models describe specific use cases for both pilots in order to validate the maturity of the uncertainty profile concepts. This comes as an update to Section 5 from D2.1, where initial test-ready models were reported. We are reporting updates to the test ready models already presented in D2.1 as well as new test ready models according to project milestones. For these test-ready models, we have followed the design considerations and methodology presented in D2.1.

4.1 FPX

The following section describes the test-ready model of the Geo Sports demonstrator of FPX, as a continuation of Section 5.2 in deliverable D2.1. At each level, we describe the use cases already presented in D2.1 where revisions were performed, together with the modeled use cases according to Milestone 3.

4.1.1 Application Level

<<The details have been removed from this document due to the agreement for confidentiality with the pilot companies>>

4.1.2 Infrastructure Level

<<The details have been removed from this document due to the agreement for confidentiality with the pilot companies>>

4.1.3 Integration Level

<<The details have been removed from this document due to the agreement for confidentiality with the pilot companies>>

4.1.4 Progress Summary

In summary we have made several progresses. Table 2 shows the progress has been achieved in comparison with D2.1. As shown, in D2.2 we have modelled several new use cases.

Table 2: Progress summary in D2.2 w.r.t. FPX use cases

Level	Use case	D2.1	D2.2
Infrastructure	UC1_INFR_1	X	
	UC1_INFR_2		X

	UC1_INFR_3		X
	UC1_INFR_4		X
	UC1_INFR_5		X
	UC1_INFR_6		X
Application	UC1_APP_9 (discarded due to use case restructuring)	X	
	UC1_APP_9_QUUPPA		X
	UC1_APP_10_QUUPPA		X
	UC1_APP_11_QUUPPA		X
	UC1_APP_15_QUUPPA		X
	UC1_APP “Rotated Locators”		X
Integration	UC1_INTE_2	x	x
	UC1_INTE_3	x	x

4.2 ULMA

The following section describes the test-ready model of the Warehouse Management System (WMS) demonstrator of ULMA, as a continuation of Section 5.3 in deliverable D2.1 [1]. At each level, we describe the use cases already presented in D2.1 where revisions were performed, together with the modeled use cases according to Milestone 3.

4.2.1 Application Level

<<The details have been removed from this document due to the agreement for confidentiality with the pilot companies>>

4.2.2 Infrastructure Level

<<The details have been removed from this document due to the agreement for confidentiality with the pilot companies>>

4.2.3 Integration Level

<<The details have been removed from this document due to the agreement for confidentiality with the pilot companies>>

4.2.4 Progress Summary

Table 3 summarizes our progress in D2.2 compared with D2.1 w.r.t. UMLA use cases modeling.

Table 3: Progress summary w.r.t. ULMA use cases

Level	Use case	D2.1	D2.2
Infrastructure	UC2_INFR_1.1	x	x
	UC2_INFR_1.2		x
	UC2_INFR_2.2		x
	UC2_INFR_2.3		x
Application	UC2_APP_1.1 (redesgined due to methodology change)	X	X
	UC2_APP_1.2		X
	UC2_APP_2.1		X
	UC2_APP_2.3		X
	UC2_APP_3.2		X
Integration	UC2_INTE_1.1	x	x
	UC2_INTE_2.2	x	x
	UC2_INTE_2.3		x

5 Summary and Future plan

This document presents the updates to existing test-ready models, to profiles, and to the modeling methodology, and the progress made with the test-ready models in WP2. These test-ready models will be enriched, and new ones will be created covering the remaining use cases, and will be presented in D2.3. The software artifact produced in this deliverable is available at [2].

5.1 Application Level

Achievement of M3 – FF

For the application level, the third milestone (M3) was fully achieved for the ULMA pilot. For 5 out of 9 use cases test-ready models were created (which represents 55%; whereas only 50% were expected). The planned score for the FPX pilot has been exceeded in M3. So far, 4 out of 6 use case have been modelled, which represents a percentage of 66% (expected 66%), with an additional use case that was identified during the technical workshop. For FPX, we decided to concentrate on the QUUPPA-related uses cases first, because they could be automated via NMT's test rig.

Future plan – FF

We plan cover 100% of the use case definitions in the next revision of D1.1 for M4 and we will report them in the future D2.3.

5.2 Infrastructure Level

Achievement of M3 – TUW

The third milestone was achieved with having updates to the profiles and models achieved in milestone 2, which allowed us to create the test-ready models promised for milestone 3. For both the Geo-sports case (FPX) and Handling Systems Demonstrator (ULMA) we have modelled delivered test-ready models, as shown in Sections 4.1.4 and 4.2.4.

Future plan – TUW

Based on our experience and results with milestone 3, we will complete the test-ready models (i.e., 100% of use-cases) and document them in D2.3.

5.3 Integration Level

Achievement of M3 – SRL

In this deliverable, according to the third milestone, modeling of 66% use cases for the both case studies at the integration level has been achieved. For the ULMA case study, in total, we have four use cases at the integration level. In this document, we presented the models of three out of four use cases, i.e., 75% of the use cases. For the FPX case study, we presented two out of three use cases, i.e., 66%. In addition, at the time of the delivery of D2.1, the actual implementation of testing APIs was not available, and thus the models from D2.1 have been updated in D2.2 to reflect the details corresponding to the implementation of testing APIs.

Future plan – SRL

After successfully achieving the results above in the third milestone, we are confident to complete the test-ready models (i.e., 100% of use-cases) for the integration level and document them in D2.3.

6 Bibliography

[1] U-Test Deliverable Report on Uncertainty Modelling Framework V.1
[2] U-Test Uncertainty Modelling Framework Artifacts, released October 2016: https://extsvnsrv.fokus.fraunhofer.de/cc/sq/UTest/browser/WP2/Deliverables/Releases/2.2
[3] Man Zhang, Shaukat Ali, Tao Yue and Phu Nguyen. Uncertainty Modeling Framework for the Integration Level V.1, Simula Research Laboratory, Technical Report 2016-01, 2016. Link: https://www.simula.no/publications/uncertainty-modeling-framework-integration-level-v1
[4] Man Zhang, Shaukat Ali, Tao Yue and Roland Norgren. An Integrated Modeling Framework to Facilitate Model-Based Testing of Cyber-Physical Systems under Uncertainty, Submitted to a Journal, Simula Research Laboratory, Technical Report 2016-02, 2016. Link: https://www.simula.no/file/sosympaperfinaltrpdf/download
[5] U-Test Deliverable Report on Requirements Collection D 1. 1
[6] U-Test Deliverable Report on Taxonomy D 1. 2
[7] http://quuppa.com, accessed on 10.10.2016.