Infrastructure-Level Uncertainties V2.0

Stefan Nastic and Hong-Linh Truong

Distributed Systems Group,
Vienna University of Technology

1 Cyber-physical systems — Infrastructure overview

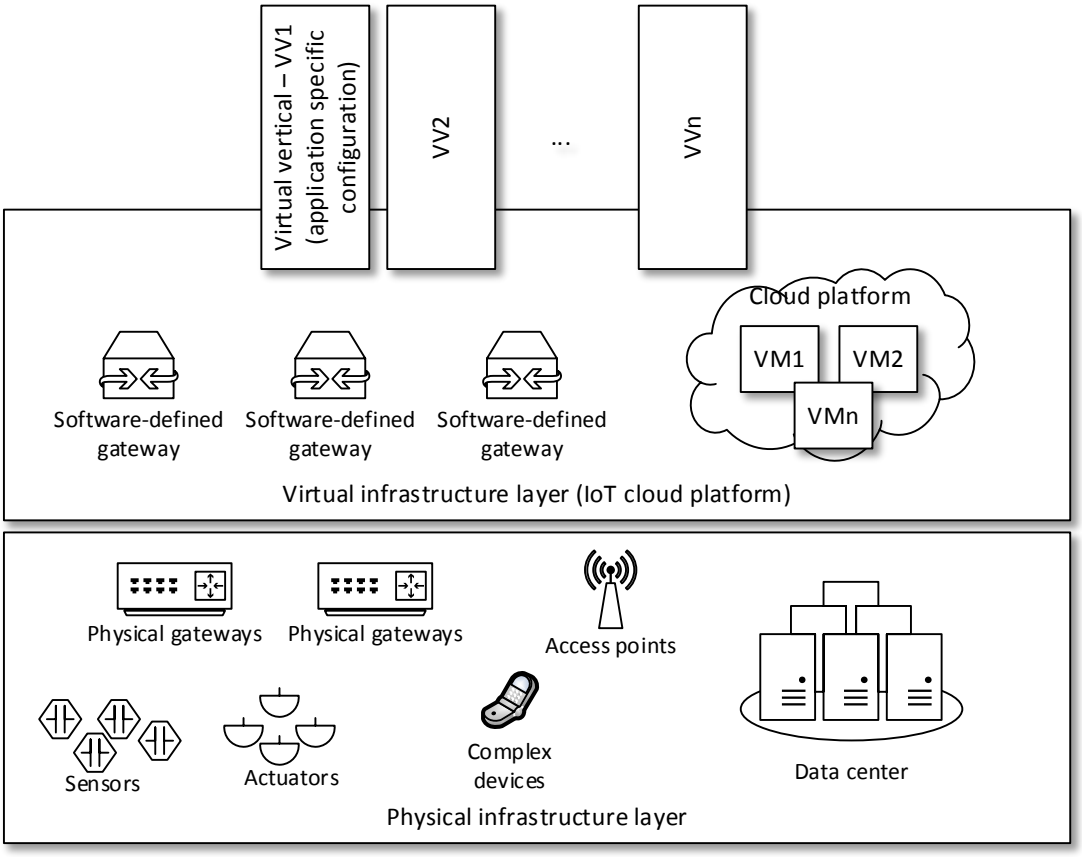

Figure 1. A high-level overview of CPS infrastructure.

The infrastructure of Cyber-Physical Systems (CPS) is complex and usually comprises variety of sensors, actuators, gateways, multiplicity of network elements and (multi) cloud platforms (e.g., VMs and cloud services). As shown in Figure 1, there are three main logical parts of CPS infrastructure: Physical layer (containing physical devices, data centers, etc.), virtual layer (encompassing virtualized CPS infrastructure, most importantly cloud platforms and software-defined gateways) and Virtual verticals (in this context application/system specific configurations and policies that directly affect CPS infrastructure).

1.1 Uncertainty states of CPS infrastructure

Generally, errors, faults and uncertain behaviors at the infrastructure level affect the execution of CPS applications independent of their 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, preprocessing 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 behavior (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 [1]. 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).

1.2 Infrastructure level uncertainties properties classes

Our taxonomy classifies the (at design time) known sources of the error and the uncertain states, e.g., the behaviors of system units which are potentially, positively or cumulatively responsible for the error/uncertain states of CPS infrastructure. Figure 2 gives an overview of the infrastructure level uncertainties taxonomy for CPS systems. The taxonomy shown in Figure 2 comprises a set of concepts (i.e., uncertainty properties classes), which are a concrete instantiations of the concepts defined in the meta model. 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 taxonomy's meta model [23].

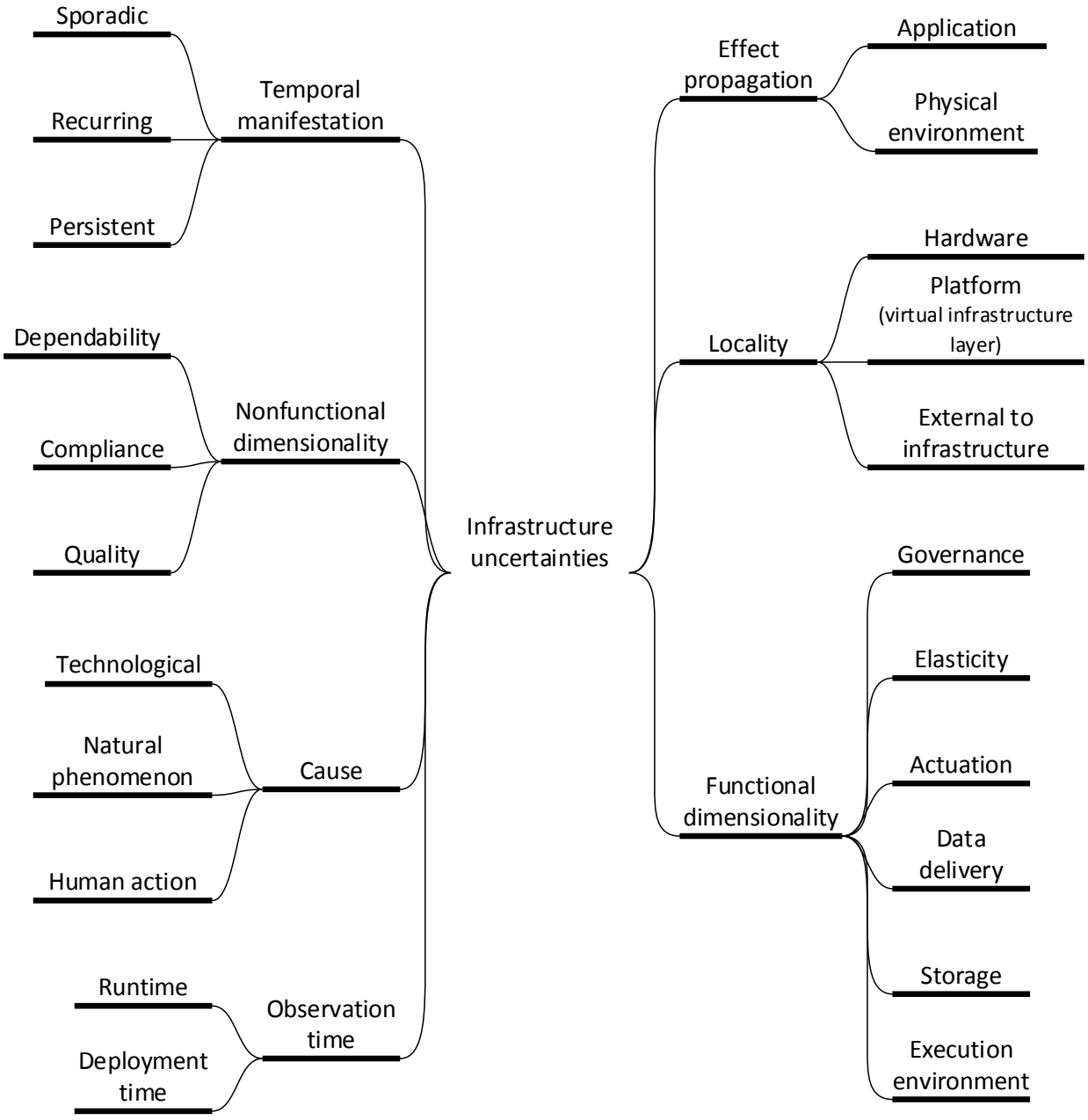

Figure 2. Infrastructure level uncertainties taxonomy for CPS systems.

1.2.1 Effect propagation (What the uncertainties affect)

Uncertainties at the CPS infrastructure can manifest themselves as failures or as functionality degradation at application level (e.g., [2]) or in the physical environment [3]. 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 conceptual model [23].

1.2.2 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) 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 [4, 5] and root cause analysis [6]. In the meta model this corresponds with the Uncertainty Location concept [23].

1.2.3 Nonfunctional dimensionality (Which nonfunctional property they affect)

The uncertainties can affect the dependability [7, 8] (e.g., safety, availability, reliability, security, etc.), data quality or legal/compliance of the CPS infrastructure [9-11]. It is worth noticing here that the nonfunctional 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 nonfunctional dimensionality of infrastructure uncertainties is derived from the conceptual model's general concepts: the Effect and the Effect Measurement [23].

1.2.4 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[12]. The Uncertainty Cause is an instantiation of Cause class from the meta model.

1.2.5 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 [13]. These concepts are instantiations of Time class from the meta model [23].

1.2.6 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 or execution environment uncertainties. The functional dimensionality is derived from state-of-the-art in CPS infrastructure research [14-22].

1.2.7 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 meta model [23].

1.3 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.

1.3.1 Data delivery uncertainties family

The data delivery uncertainties family includes such uncertainties that affect the infrastructure's facilities for generating, preprocessing 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 *legal/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 3[1]

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 3

[1] *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 delivery 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 3

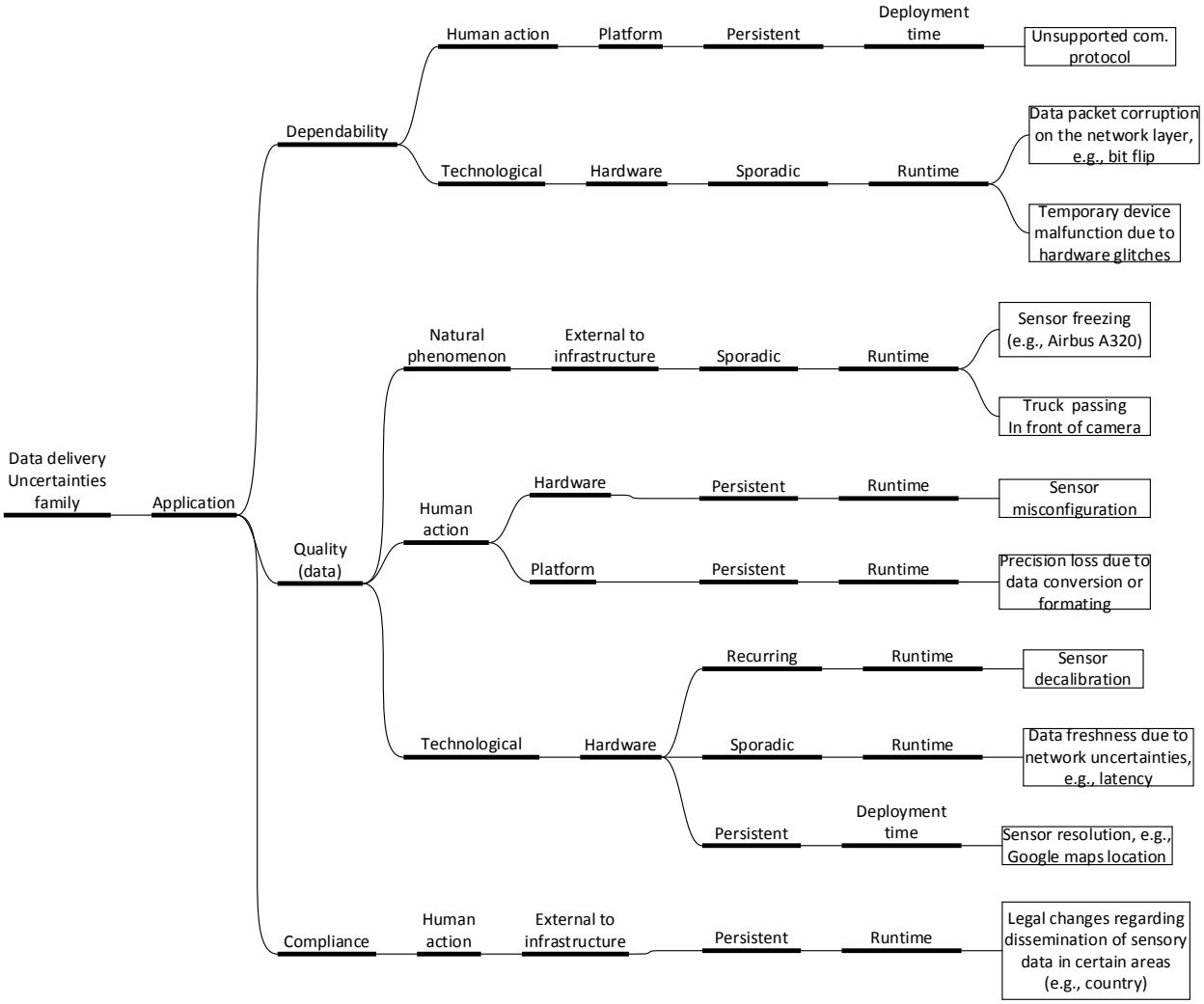

Figure 3. Data delivery uncertainties family.

1.3.2 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 application runtime. The actuation uncertainties family comprises three main elementary categories: Actuation legal/compliance uncertainties, actuation uncertainties affecting the *dependability of applications* and actuation uncertainties affecting the *dependability of environment.*

Name:	Actuation 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 4

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 4

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 4

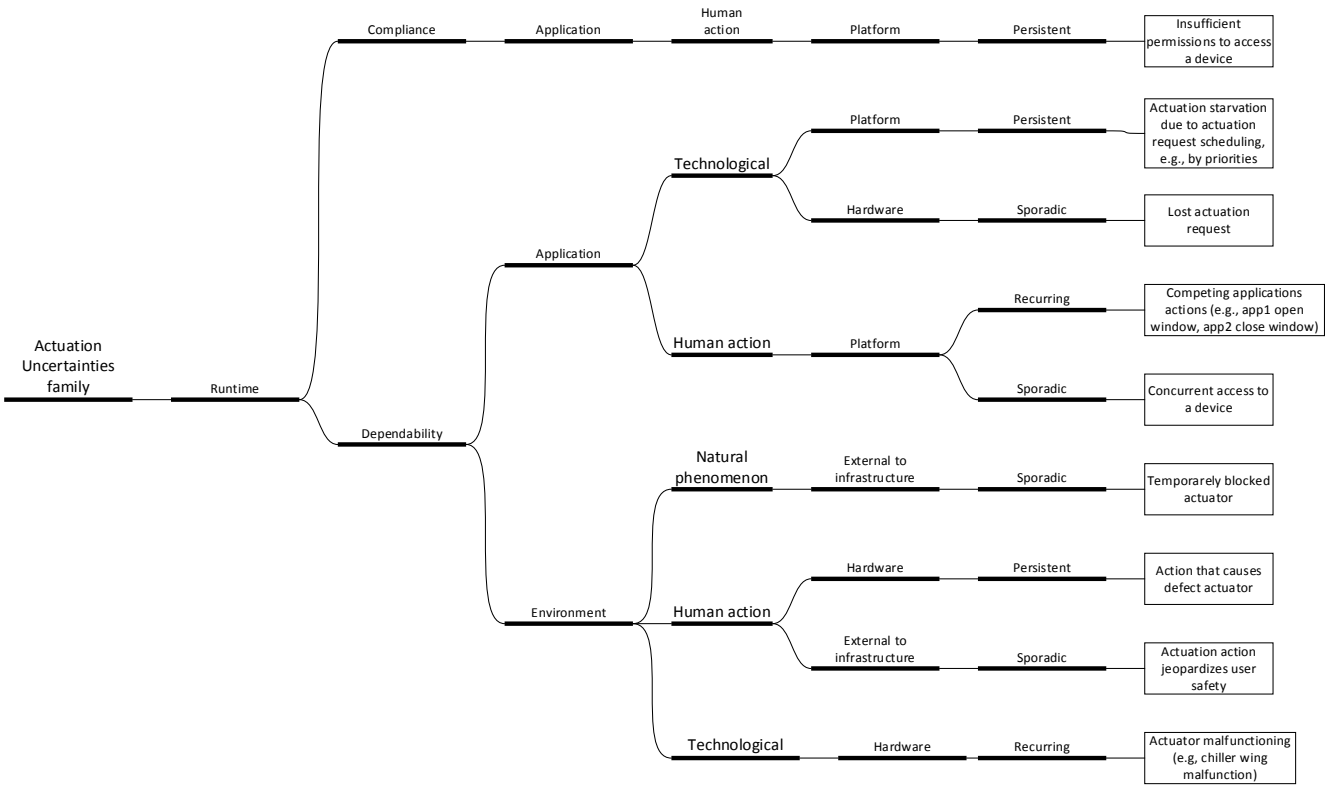

Figure 4. Actuation uncertainties family

1.3.3 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.*

Name:	Deployment time execution environment uncertainties
Definition:	These uncertainties are observed during application's deployment phase. The nonfunctional 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 5

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 5

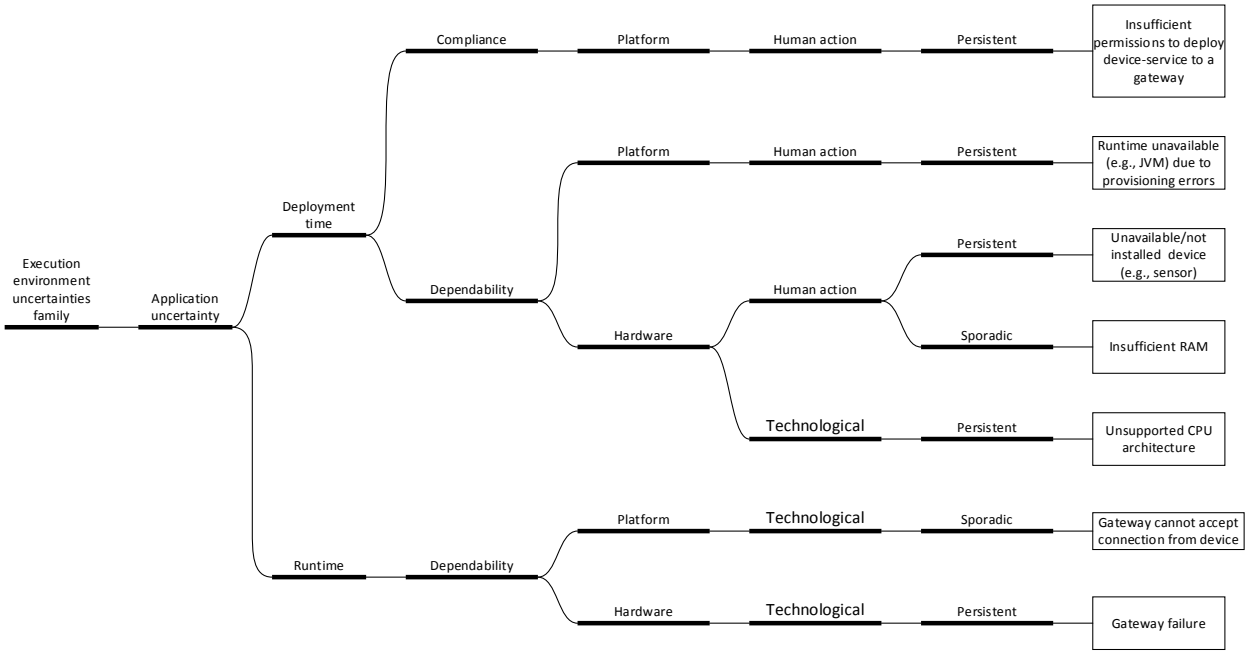

Figure 5. Execution environment uncertainties family.

1.3.4 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 6

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 6

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 6

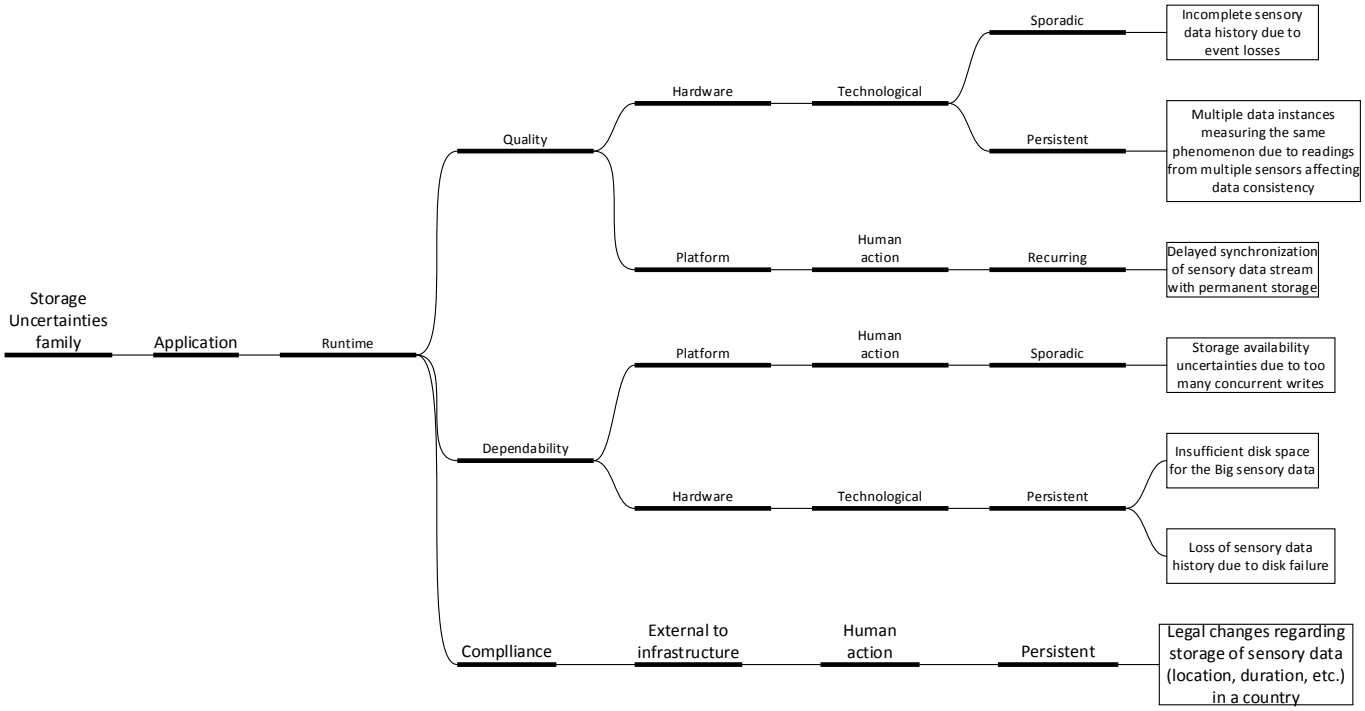

Figure 6. Storage uncertainties family.

1.3.5 Elementary uncertainties families – aggregated view

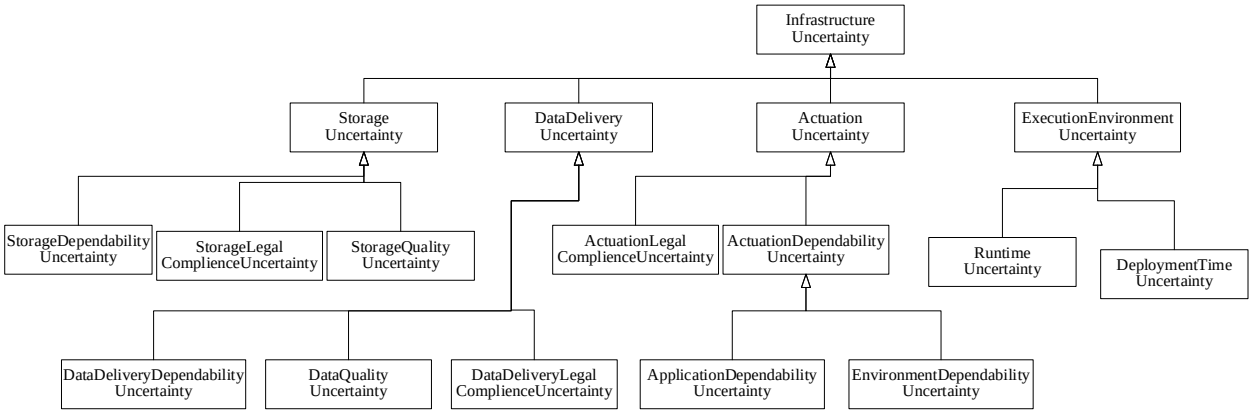

Figure 7. UML diagram showing the elementary uncertainties families.

1.4 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 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.

1.4.1. 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 8 shows UML diagram of the composed uncertainties families related to governance uncertainties.

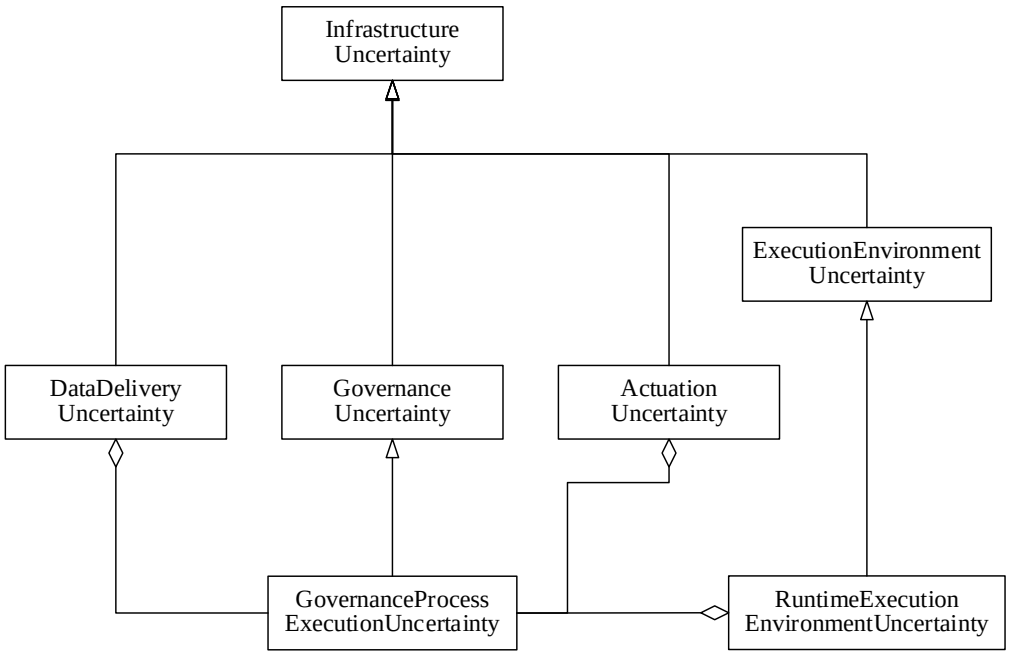

Figure 8. Governance uncertainties families (partial view).

1.4.2. 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 behavior of the application. 2. Information data source (e.g., a WebServer reporting response time), is overloaded or crashes, and does not report data anymore. This can generate two problems, depending on the behavior 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 behavior information not being available.

Name:	Cloud Service behavioral 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 Bandwith, 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 9 shows UML diagram of the composed uncertainties families related to elasticity uncertainties.

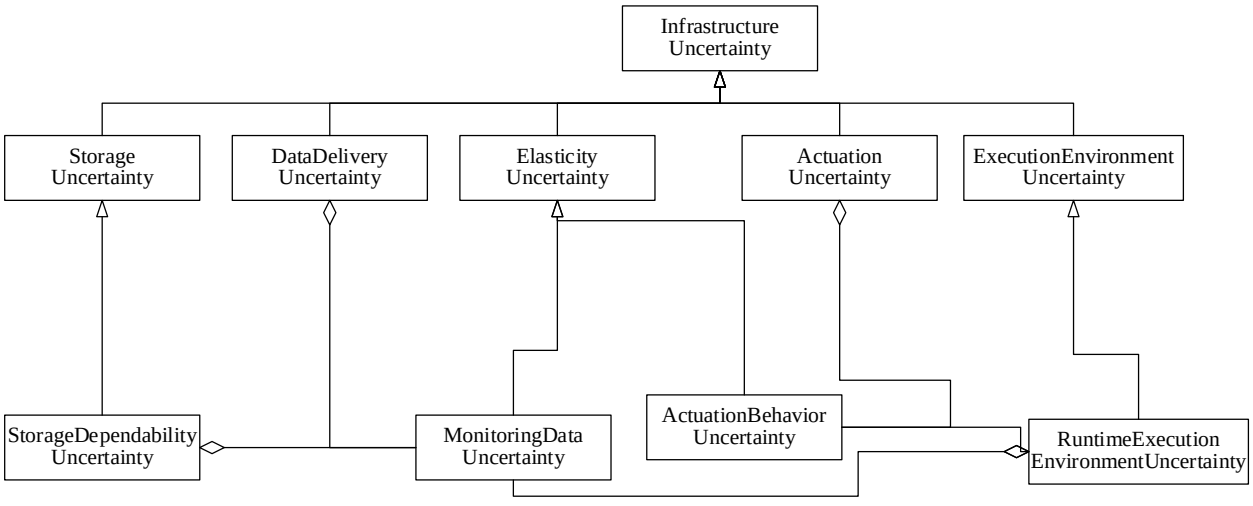

Figure 9. Elasticity uncertainties families (partial view).

1.5 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.

References

[1] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," *Dependable and Secure Computing, IEEE Transactions on*, vol. 1, no. 1, pp. 11-33, 2004.

[2] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec, "The many faces of publish/subscribe," *ACM Computing Surveys (CSUR)*, vol. 35, no. 2, pp. 114-131, 2003.

[3] M. M. N. Magazin. "m2m now," March 2015; http://www.m2mnow.biz/2014/08/05/23417-employee-safety-security-regulations-raise-stakes-fleet-operators/.

[4] T. Aslam, I. Krsul, and E. H. Spafford, "Use of a taxonomy of security faults," 1996.

[5] M. łgorzata Steinder, and A. S. Sethi, "A survey of fault localization techniques in computer networks," *Science of computer programming*, vol. 53, no. 2, pp. 165-194, 2004.

[6] M. Leszak, D. E. Perry, and D. Stoll, "A case study in root cause defect analysis." pp. 428-437.

[7] A. Avizienis, J.-C. Laprie, and B. Randell, *Fundamental concepts of dependability*: University of Newcastle upon Tyne, Computing Science, 2001.

[8] "ISO/IEC 25010:2011," May 2015; http://www.iso.org/iso/catalogue_detail.htm?csnumber=35733.

[9] D. M. Strong, Y. W. Lee, and R. Y. Wang, "Data quality in context," *Communications of the ACM*, vol. 40, no. 5, pp. 103-110, 1997.

[10] T. Buchholz, A. Küpper, and M. Schiffers, "Quality of context: What it is and why we need it."

[11] R. H. Weber, "Internet of things–Governance quo vadis?," *Computer Law & Security Review*, vol. 29, no. 4, pp. 341-347, 2013.

[12] S. Plato. "Stanford Plato Phenomenolgy," http://plato.stanford.edu/entries/phenomenology/.

[13] J. Gray, "Why do computers stop and what can be done about it?," *Symposium on reliability in distributed software and database systems.* pp. 3-12.

[14] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787-2805, 2010.

[15] M. Yuriyama, and T. Kushida, "Sensor-cloud infrastructure-physical sensor management with virtualized sensors on cloud computing." pp. 1-8.

[16] P. Stuedi, I. Mohomed, and D. Terry, "WhereStore: Location-based data storage for mobile devices interacting with the cloud." p. 1.

[17] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for vm-based cloudlets in mobile computing," *Pervasive Computing, IEEE*, vol. 8, no. 4, pp. 14-23, 2009.

[18] E. Cuervo, A. Balasubramanian, D.-k. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "MAUI: making smartphones last longer with code offload," *Proceedings of the 8th international conference on Mobile systems, applications, and services.* pp. 49-62.

[19] K. Bhardwaj, S. Sreepathy, A. Gavrilovska, and K. Schwan, "ECC: Edge Cloud Composites," *In 2nd IEEE International Conference on Mobile Cloud Computing, Services, and Engineering* pp. 38-47.

[20] S. Nastic, S. Sehic, D.-H. Le, H.-L. Truong, and S. Dustdar, "Provisioning Software-defined IoT Cloud Systems," *In 2nd International Conference on Future Internet of Things and Cloud (FiCloud)* pp. 288-295.

[21] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," *In 1st MCC workshop on Mobile cloud computing*pp. 13-16.

[22] S. Nastic, M. Vögler, C. Inzinger, H.-L. Truong, and S. Dustdar, "rtGovOps: A Runtime Framework for Governance in Large-scale Software-defined IoT Cloud Systems,". *In 3rd IEEE International Conference on Mobile Cloud Computing, Services, and Engineering.* p. G5.

[23] M. Zhang, S. Ali, T. Yue, D. Pradhan, B. Selic, O. Okariz, and R. Norgren, “An Uncertainty Taxonomy to Support Model-Based Uncertainty Testing of Cyber-Physical Systems,” tech. rep., Simula, 2015.