2018 IEEE International Conference on Smart Computing

Reasoning about Smart City

Marcello Balduccini*, Edward Griffor†, Michael Huth‡, Claire Vishik, Martin Burns†, David Wollman†

*Saint Joseph's University
Philadelphia, PA, USA
Email: marcello.balduccini@sju.edu

†US NIST
Gaithersburg, MD, USA
Email: {edward.griffor,martin.burns,david.wollman}@nist.gov

‡Imperial College London
London, UK
Email: m.huth@imperial.ac.uk

§Intel Corporation
Austin, TX, USA
Email: claire.vishik@intel.com

***Abstract*—Smart Cities are complex environments, comprising diverse cyber-physical systems (CPS), including Internet of Things (IoT). Smart Cities pose challenges of scale, integration, interoperability, sophisticated processes, governance, human elements. Trustworthiness (including safety, security, privacy, reliability and resilience) of these Smart Cities and their elements is critical for gaining broad adoption by the leadership and the public. The US National Institute of Standards and Technology (NIST) and its government, university and industry collaborators, have developed an approach to reasoning about CPS/IoT trustworthiness that can be applied to Smart Cities. The approach uses ontology and reasoning techniques, is based on the NIST Framework for Cyber-Physical Systems, and demonstrates how a greater understanding of the interdependencies between concerns (elements of the CPS Framework) can be achieved. To demonstrate capabilities of the approach in a short paper, we develop a public safety use case and show how reasoning can be used to analyze and validate the trustworthiness of elements of Smart Cities.**

I. Introduction

A *Smart City* can be described as a system of systems, including instances of Cyber-Physical Systems (CPS)/Internet of Things (IoT), that exhibits both scale and horizontal integration. "Scale" does not merely refer to a relatively large set of CPS. It also includes subsets that target a dynamic but coordinated function, and describe the interaction of these subsets to realize the goals set out in making a given city *smart*. As such they exhibit *complexity* in technological diversity and also emergent diversity in usage of their functions.

Horizontal integration – e.g. between street lights, water pumps, and traffic sensors – needs to reflect a diversity of ownership and technology of components of a Smart City. There is no sole horizontal platform that can span all aspects of a Smart City, and if such a platform were to exist, it could form a single point of failure or present additional cyber risks.

The analysis of complex systems is improved by using specialized models or frameworks, and such frameworks exist for many areas applicable to Smart Cities, e.g., the NIST CPS Framework. The complexity of Smart Cities creates a strong need for knowledge representation and reasoning tools, but this has not yet received sufficient attention.

The use of ontologically inspired modeling in computer science is not new. As Smith and Welty [1] point out, this approach has been used extensively in information systems science. Examples include conceptual modeling in the database development area or domain modeling in software engineering.

The creation of an extensive ontology is frequently a lengthy process. However, existing frameworks, such as the CPS Framework, can speed up ontology development for an area, thus creating premises for faster emergence of reasoning and decision support applications. In this case, the authors had the advantage to rely on an extensive model already in existence. NIST hosted a Public Working Group on CPS with the aim of capturing input from those involved in CPS to define a *CPS reference framework* supporting common definitions and facilitating interoperability between such systems. A key outcome of that work is the CPS Framework (Release 1.0) [2], which proposes a means of describing three *facets* during the life of a CPS: conceptualization, realization, and assurance of CPS;

978-1-5386-4705-9/18/$31.00 ©2018 IEEE
DOI 10.1109/SMARTCOMP.2018.00033

and to facilitate these descriptions through analytical lenses, called *aspects*, which group common concerns addressed by the builders and operators of the CPS. In the framework, the aspect named Trustworthiness describes multiple related *concerns* that deal specifically with the avoidance of harm in Privacy, Security, Safety, Resilience, and Reliability. The framework is extensible and supported with additional models, e.g. a UML model of concerns, aspects, all three facets, and the interdependencies across the CPS lifecycle.

The CPS Framework articulates the artifacts of a CPS in a precise way, including the concerns that motivate important requirements to be considered in conceptualizing, realizing (including operating), and assuring CPS. However, the CPS Framework does not, by itself, have the ability to reason over the CPS lifecycle. In this short paper, we propose to use ontology-based reasoning to realize such capabilities and illustrate this through a case study relevant to Smart Cities that focuses on the Trustworthiness aspect: the Ontology is used to model the CPS operations from scenarios associated with an advanced body camera for public safety personnel.

The model contains sufficient complexity to demonstrate the capabilities of the approach and its applicability to smart-city infrastructures. The case study includes, e.g., considerations such as Transduction (where a CPS produces a physical signal that interacts with the Environment) and Influence (where a CPS produces or receives a physical signal causing a state change of another CPS).

II. Related Work: Ontology-Based Reasoning

Recognizing the complexity of a Smart City, researchers have started focusing on efforts that include ontologies and reasoning. Among notable research papers published recently, we can mention an ontology for energy management in Smart Cities [3] or an ontology focusing on Smart City knowledge exploitation [4], [5]. Ontologies for security and trustworthiness have also been pursued by researchers in recent years. Examples include work on ontologies for certification and testing [6], and work on ontology integration in security [7].

However, there has been no work yet on ontology-based reasoning for the Trustworthiness aspect of Smart Cities.

III. A CPS Reference Framework

The NIST CPS Framework provides the taxonomy and methodology for conceptualizing, realizing, and assuring cyber-physical systems that meet the expectations and concerns of system stakeholders, including engineers, users, and the community that benefits from the system's functions. The Framework comprises a set of concerns about systems, three development facets and a notion of functional decomposition suited to CPS. A CPS often delivers complex functions that are ultimately implemented in diverse inter-operating systems and devices. Interactions can occur through the exchange of information or the exchange of matter/energy. The former are *logical* interactions and the latter *physical* interactions.

The functional decomposition of a CPS breaks it down, from a name and brief description of what the system is or does – the *Business Case* – through the set of scenarios or step-by-step descriptions of ways of using the system and the functions that realize those steps – the *Use Case* – the actors/subsystems and interactions – the *Allocation of Function* – and to the allocation of given subsystem functions to physical or logical implementation – *Physical-Logical Allocation*.

Concerns about CPS/IoT are represented in a forest, where trees and branching corresponds to the *decomposition of concerns*; see Fig. 1 for a view of the Trustworthiness *concern tree*. We refer to this structure as the *concern forest* of the CPS Framework. The concerns at the roots of this structure, the highest level concerns, are called *aspects*; there are nine aspects, one of which being Trustworthiness.

A concern about a given system reflects a dimension of issues to be addressed in the realization of a CPS. A *property* is a requirement or statement that addresses a concern. This method or practice is applied to each function in the functional decomposition of the system. A concern can be uniquely identified with a branch in the concern forest, and can be represented as consisting of a root followed by a (possibly empty) sequence of concern names in the branch, separated by dots. In the Trustworthiness aspect, e.g., we have the concern $Trustworthiness.Security.Cybersecurity.Confidentiality$, which may be abbreviated as, e.g., $Conf'd$. A sample property, meant to address this concern about data exchanged between components of a system, is the use of encryption of some kind (e.g. AES). A property is appended to the concern tree branch in block parentheses. For instance, $Conf'd[AES\text{-}encr]$ states that concern $Conf'd$ is intended to be addressed by the use of AES encryption.

The Framework provides guidance on forming an *Assurance Case for each concern applied to the CPS*, comprised of: properties of the CPS and the concerns that resulted in the addition of those properties to the model of the CPS; *argumentation* or criteria for concluding that a property has been established of the CPS; *evidence* information, accessible to stakeholders, that the criteria used in this argumentation are indeed met; and *uncertainty* associated with the evidence that the criteria are met. The framework has been applied to complex environments including CPS, such as Smart Cities, and provides a structured way to analyze complex environments. It can be argued that the framework and the Open Source technology, depicted in Figure 2, are essential to understanding critical performances of CPSs incrementally, from the perspective of CPS development, deployment, and adoption.

IV. A CPS Framework Ontology

In order to develop reasoning capabilities for the CPS framework, we developed an ontology of the CPS Framework [8]. An ontology is a formal, logic-based representation of knowledge supporting reasoning by means of logical inference. In this paper, we adopt a broad view of this term: by "ontology", we mean a collection of statements in a logical language that represent a given domain in terms of *classes* (i.e., sets) of objects, *individuals* (i.e., specific objects), relationships

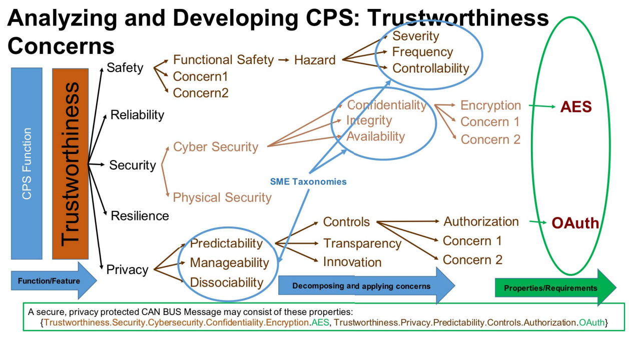


Fig. 1: Branching for Trustworthiness Concerns

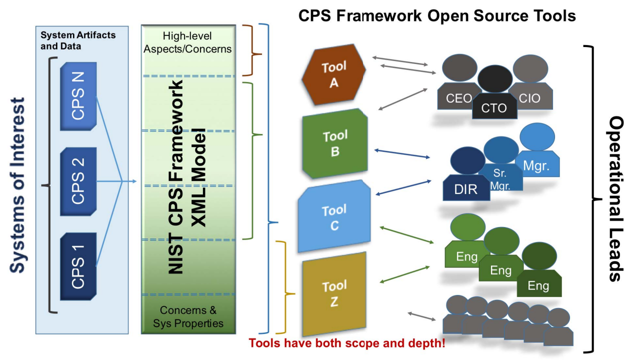

CPS Framework Open Source Tools
Systems of Interest
System Artifacts and Data
CPS N
CPS 2
CPS 1
High-level Aspects/Concerns
NIST CPS Framework XML Model
Concerns & Sys Properties
Tool A
Tool B
Tool C
Tool Z
CEO
CTO
CIO
DIR
Sr. Mgr.
Mgr.
Eng
Eng
Eng
Operational Leads
Tools have both scope and depth!

Fig. 2: Dashboards using the CPS Framework Open Source Tools

between objects and/or classes, and logical statements concerning these relationships. For example, an ontology focusing on the trustworthiness of CPS may define the high-level concept of "Concern" with its refinement of "Aspect." All of these are formalized as classes and, for Aspect, subclasses. Specific concerns are represented as individuals: $Trustworthiness$ as an individual of class Aspect, $Security$ and $Cybersecurity$ as individuals of class Concern. Also, a relation "has-subconcern" associates a concern with its sub-concerns. Thus, Aspect "has-subconcern" $Security$, which in turn "has-subconcern" $Cybersecurity$. By introducing a property "satisfied," one can also indicate which concerns are satisfied.

Inference can then be applied to propagate "satisfied" and other relevant properties and relations through the ontology. For example, given a concern that is not "satisfied," one can leverage relation "has-subconcern" to identify other concerns that are not satisfied because of it, either directly or indirectly.

In practice, it is often convenient to distinguish between the factual part, Ω, of an ontology and the *axioms*, Λ. The former, from now on simply called "ontology," encodes the factual information, e.g., $Trustworthiness$ "has-subconcern" $Security$. The latter expresses deeper, often causal, links between relations, e.g. that a concern is not satisfied if any of its sub-concerns is not satisfied. Further, when discussing reasoning tasks, we will also indicate, separately, the set $\mathcal{Q}$ of axioms encoding a specific reasoning task or query.

V. Applying Ontology and Reasoning to CPS

Our approach to reasoning leverages a logic-based representation of a system of interest and applies inference to draw new and useful conclusions in a rigorous way. It is agnostic to specific choices of logical language and inference mechanism. It assumes the existence of axioms in the selected logical language, which formalize the queries one is interested in answering, the type of reasoning to be carried out, and any contextual information. Conclusions are drawn from an ontology Ω and a set of axioms Λ by means of a logical inference procedure, denoted by symbol $\vdash$. If Δ follows from Ω and Λ, we write $\Omega \cup \Lambda \vdash \Delta$, where $\cup$ denotes set union.

For example, in the context of cybersecurity, the language of *propositional logic* can be used to represent (a) that a cyberattack occurred (statement p) and (b) expert knowledge that, when that cyberattack occurs, a certain system becomes inoperative (statement $p \supset q$, read "p implies q," where q states that the system is inoperative). The logical inference $\{p\} \cup \{p \supset q\} \vdash q$ allows one to draw the conclusion that q holds, i.e. that, as a result of the cyberattack, the system is expected to be inoperative.

A. Formalization

For the sake of illustration of how systems forming Smart Cities are analyzed, we consider a use case centered around an advanced model of body camera (BCAM) for safety personnel. The BCAM system comprises a camera and a situational awareness module (SAM). The SAM controls the camera's configuration and processes the video stream received from it, which is then made available, through a physical output, to a third party. The camera and the SAM may use encrypted memory and secure boot. The third party that receives the output of the BCAM system, e.g. a court of law, will reject it if issues are detected in the feed. This use case is chosen because it encompasses major component types of a CPS, and lends itself to various non-trivial investigations. It also denotes a typical activity in a Smart City, in this case, sharing information for safety purposes. Through this use case, we will highlight the interplay among trustworthiness concerns, as well as their ramifications on other CPS aspects, such as the functional aspect.

For the sake of simplicity, we assume that the camera is capable of two recording modes, one at 25 frames per second (fps) and the other at 50 fps. The selection of the recording mode is made by the SAM, by acting on a flag of the camera's configuration. It is assumed that two camera models exist, a basic one and an advanced one. Either type of camera can be used when realizing the CPS. Due to assumed technical limitations, the basic camera is likely to drop frames if it attempts to record at 50 fps while using encrypted memory.

In our approach, the formalization of a CPS is organized along multiple levels: (L1) aspects and concerns; (L2) properties; (L3) CPS configuration; (L4) actions; (L5) constraints, dependencies and trade-offs; and (L6) satisfaction axioms. Level L1 and L6 form the *CPS-independent specification*, since aspects and concerns are independent of the specific CPS

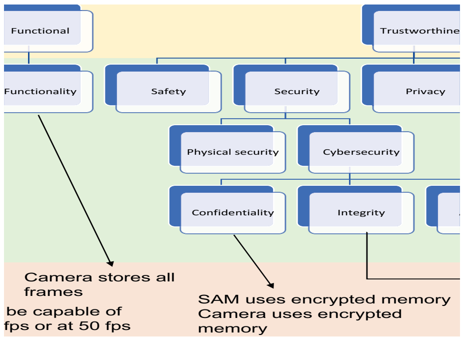


Fig. 3: BCAM use case: fragment of the concern forest

being modeled. Levels L2-L5 comprise the *CPS-dependent specification*, as the information included in them depends on the CPS being modeled. From another perspective, levels L1 and L2 formalize the concepts from the definition of the CPS Framework. Levels L3-L5 extend the framework to provide details needed for reasoning about the behavior of a CPS of interest. Level L6 provides the semantics of the formalization.

Formalization of aspects and concerns. The formalization of aspects and concerns is shared by all CPSs. The nodes of a concern tree are represented by individuals of class *Concern*. The root nodes of the concern trees are a particular kind of concern, and so they are placed in a class (*Aspect*) that is a subclass of *Concern*. Following the definition of the CPS Framework, class *Aspect* includes individuals *Trustworthiness*, *Timing* and *Functional* for the corresponding aspects, while class *Concern* includes individuals *Security*, *Cybersecurity*, *Functionality*, etc.

Edges linking aspects and concerns are represented by the relation subConc, which is a representation of the notion of sub-concern. Thus, an edge from a concern x to a concern y is formalized by a statement $\mathsf{subConc}(x, y)$. Statement $\mathsf{subConc}(\mathit{Trustworthiness}, \mathit{Security})$, e.g., formalizes that the Security concern is a direct sub-concern of the Trustworthiness aspect in our BCAM use case. Concerns $\mathit{Cybersecurity}$ and $\mathit{Conf'd}$ are linked similarly.

Formalization of properties. Properties of a CPS are represented by individuals of class *Property*. An edge that links a property with an aspect or concern is represented by relation addrBy, which stands for "addressed by." Let us suppose that, in the BCAM use case, both SAM and camera must use encrypted memory for the confidentiality concern to be satisfied (see Figure 3). We may express this by two statements $\mathsf{addrBy}(\mathit{Conf'd}, \mathit{SAM\_mem}[\mathit{encr}])$ and $\mathsf{addrBy}(\mathit{Conf'd}, \mathit{cam\_mem}[\mathit{encr}])$. Similarly, the fact that SAM and camera must use secure boot for the integrity concern to be satisfied is expressed by $\mathsf{addrBy}(\mathit{Integrity}, \mathit{SAM\_boot}[\mathit{sec}])$ and $\mathsf{addrBy}(\mathit{Integrity}, \mathit{cam\_boot}[\mathit{sec}])$.

Another property, used below, is $\mathit{cam}[\mathit{storeAll}]$, stating that camera cam stores all frames, i.e. does not drop any frames. Note that, in the BCAM use case, the court's decision to admit or reject the feed depends on the quality of the feed: not dropping any frames is essential for ensuring the satisfaction of the CPS' functionality concern and, consequently, admissibility in court. Also note that the example of admissibility in court is used for sake of illustration only.

Statement type	**Syntax**
Property dependency	• Γ impacts$_{\text{pos}}$ π • Γ impacts$_{\text{neg}}$ π
Default property value	• σ defaults true • σ defaults false
Effects of actions	• a causes π if Γ
Triggered actions	• Γ triggers a

TABLE I: Constraints, dependencies, and trade-offs where Γ, π range over (sets of) propositions and a over actions

Formalization of configurations. Properties do not necessarily capture all possible configurable features of a CPS, but only those on which concerns are defined. For instance, in the BCAM use case, there is a choice between using the basic camera or the advanced camera. We describe the choice between the two as part of the configuration of the CPS. Thus, the formalization includes a class *Configuration*. Each individual of this class represents a different configuration feature, e.g., $\mathit{cam}[\mathit{basicOne}]$ is used for the selection of a type of camera cam. Truth values of properties and configurations are specified by relation obs, where a statement $\mathit{obs}(x, \mathit{true})$ declares that property or configuration x is (observed to be) true. Observability of falsity is represented in a similar way.

Formalization of actions. We use the term "action" to denote both those actions that are within the control of an agent (e.g., actions a driver may take), and those actions that occur spontaneously, e.g. triggered by a particular state of the CPS – such as rejecting the feed produced by the BCAM system if the camera malfunctions. The formalization includes a suitable class *Action* and individuals for the actions of interest. In the BCAM use case, we consider the occurrence of a cyberattack, and formalize it by means of the individual/action labeled *Attack*. The case in which the court rejects the feed is modeled by an individual $\mathit{RejectFeed}$. When the configuration of a CPS can be modified at run-time, suitable actions $MakeTrue(c)$ and $MakeFalse(c)$ may also be introduced, where c is the configuration the action affects. For example, in the BCAM use case, we consider actions $MakeTrue(\mathit{cam}[\mathit{basicOne}])$ and $MakeFalse(\mathit{cam}[\mathit{basicOne}])$, which enable, respectively, the basic camera or the advanced camera.

Formalization of constraints, dependencies, trade-offs. One can establish causal links between concerns, properties, configurations, and actions. This is accomplished by the statements shown in Table I, where they are grouped and listed with their syntactic expressions as judgments. Encodings of each statement for a back-end reasoner that implements reasoning capabilities are also provided (omitted in this short paper).

Consider that the use of encrypted memory causes the basic camera to drop frames if it attempts to record at 50 fps. We formalize this by the property dependency statement:

$$\mathit{cam\_mem}[\mathit{encr}] \wedge \neg \mathit{cam}[\mathit{rate25fps}] \wedge \mathit{cam}[\mathit{basicOne}] \quad \text{impacts}_{\text{neg}} \ \mathit{cam}[\mathit{storeAll}] \tag{1}$$

The statement expresses that, under the conditions specified, the $storeAll$ property is *impacted negatively*, i.e., it is made false. If a property is impacted positively, $\mathsf{impacts_{pos}}$ is used instead. As shown in this example, properties and configurations can be negated by prefixing them by symbol $\neg$. In the case of $storeAll$, one may also want to specify that the property should be assumed to hold true in the absence of contrary evidence. This can be achieved by a statement:

$$storeAll \ \mathsf{defaults} \ true$$

The effects of actions on properties are given by statements borrowed from action language $\mathcal{AL}$ [9], which has been designed specifically for a compact specification of the causal dependencies in complex domains. For example, in the BCAM use case a cyberattack may force the camera to record at 50 fps. Using action *Attack*, introduced earlier, this may be formalized by the statement:

$$Attack \ \text{causes} \ \neg rate25fps.$$

The last type of statement from Table I describes the spontaneous triggering of actions when suitable conditions are satisfied. To illustrate this, recall that, in the BCAM use case, the feed could be rejected by the court if issues are detected in the input received from the SAM. One obvious circumstance in which this will happen is if the system is not fully functional. This link can be formalized by the trigger:

$$\neg Functional \ \text{triggers} \ RejectFeed. \tag{2}$$

Axioms. Recall that our approach reduces the task of answering a query of interest to that of finding one or more answers, Δ, such that $\Omega \cup \Lambda \vdash \Delta$ holds, where the ontology Ω and any supporting axioms Λ are expressed in a logical language for the reasoner of choice. The statements presented so far can be translated into most logic languages. Set Λ also includes statements, needed to support the reasoning tasks, that are independent of the particular CPS being modeled. The most important of such statements is:

A concern is satisfied when all of the properties that address it and all of its sub-concern are satisfied. (3)

Note that axiom (3) is responsible for recursively propagating the satisfaction of properties and concerns, or lack thereof, up the relevant concern tree, based on the information from statements addrBy and $\mathsf{subConc}$ of the ontology Ω. Thus, if the basic camera is used with encrypted memory while recording at 50 fps, (1) makes it possible to conclude that property $storeAll$ is not satisfied, while (3) yields that the *Functionality* concern and the functional aspect are not satisfied.

B. Reasoning

Next, we illustrate how this formalization may be used to reason about aspects and concerns of a CPS and their interdependencies, in relation to other systems. Most of the reasoning tasks are centered on determining whether a certain statement p is true in a state s (also called a *time step*), which is represented by an expression of the form $holds(p, s)$.

Concern tree. For the BCAM CPS, let the basic camera be used, SAM and camera use encrypted memory and secure boot, and the recording rate be set to 50 fps. Once aspects, concerns, properties, and configurations are formalized as described earlier, this system state is formalized by statements:

$$obs(basicOne, true), \ obs(cam\_mem[encr], true), \\ obs(cam\_boot[sec], true), \ obs(cam[rate25fps], false), \\ obs(SAM\_mem[encr], true), \ obs(SAM\_boot[sec], true)$$

By inspecting Figure 3, it is not difficult to see that the confidentiality concern is satisfied. From a technical perspective, a query "is χ satisfied by the design of the CPS?", where χ is a property (e.g., $storeAll$) or concern, is answered by checking whether $\Omega \cup \Lambda \vdash holds(\chi, 0)$. By specifying a different time step, one can also check whether the query is satisfied at run-time. In our running example, starting from the observation that encrypted memory is used, axiom (3) allows one to conclude that $\Omega \cup \Lambda \vdash holds(sat(Conf'd), 0)$. Similarly, one can formally conclude $holds(sat(Integrity), 0)$. From (3), it also follows that *Cybersecurity* is satisfied and, in turn, all concerns up to *Trustworthiness*. Thus the BCAM CPS is deemed to be trustworthy.

On the other hand, $\Omega \cup \Lambda$ yields $\neg holds(storeAll, 0)$, i.e. $storeAll$ is false at 0 , and so $\neg holds(sat(Functional), 0)$ is true. Recursively, the *Functionality* concern and the *Functional* aspect are not satisfied.

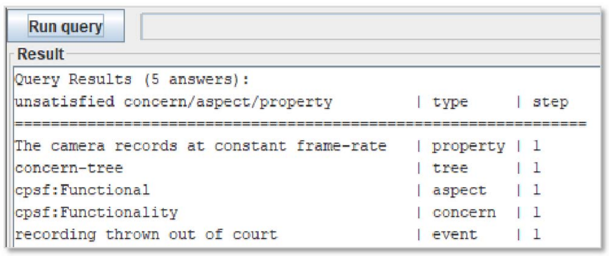
All-sat. One may also want to check whether all aspects are satisfied. This query is encoded by a set $\mathcal{Q}$ that contains

$$sat(all) \ \mathsf{defaults} \ true. \tag{4}$$

together with axioms defining the "meta-aspect" all and the fact that $sat(all)$ is true if-and-only-if the entire concern forest is satisfied. In our example, one can check that $\Omega \cup \Lambda \cup \mathcal{Q} \vdash \neg holds(sat(all), 0)$. In fact, as we saw above, $\neg holds(sat(Functionality), 0)$ is true. Thus, the CPS is deemed to be trustworthy, but does not satisfy the functional aspect. The concern forest, as a whole, is thus not satisfied.

What-if. A *What-if* reasoning task studies how the CPS is affected by the occurrence of actions, in terms of which properties hold, which concerns are satisfied, and which other actions may be triggered. Let the expression $occurs(a, s)$ denote the occurrence of action a at step s and let a history $\mathcal{H}$ be a set of such expressions. A query "is χ satisfied at step s'?", where χ is a property (e.g., $storeAll$) or concern and s' is a step during or after history $\mathcal{H}$, is answered by checking whether $\Omega \cup \Lambda \cup \mathcal{H} \vdash holds(\chi, s')$.

A query "does action a occur at step s'?" is answered by checking whether $\Omega \cup \Lambda \cup \mathcal{H} \vdash occurs(a, s')$. The same mechanism allows for answering more general questions, such as "is/isn't χ satisfied at some point during $\mathcal{H}$?" and "which actions are triggered during $\mathcal{H}$?". In reference to the BCAM use case, let us consider a scenario where initially the basic camera is used, SAM and camera use encrypted memory and secure boot, and the recording rate is set to 25 fps. Here, the functional aspect is satisfied. We can study if the functional aspect is satisfied after $\mathcal{H} = \{occurs(Attack, 0)\}$ by checking



```
Query Results (5 answers):
unsatisfied concern/aspect/property         | type     | step
==============================================================
The camera records at constant frame-rate   | property | 1
concern-tree                                | tree     | 1
cpsf:Functional                             | aspect   | 1
cpsf:Functionality                          | concern  | 1
recording thrown out of court               | event    | 1
```

Fig. 4: Implemented prototype: *What-if* reasoning task

whether $\Omega \cup \Lambda \cup \mathcal{H} \vdash holds(sat(Functional), 1)$. Note the use of step 1, which corresponds to the step following the hypothesized occurrence of the action. Intuitively, the attack forces the camera to record at 25 fps. Since our model captures (1), it follows that the camera will begin to drop frames, which in turn affects the functional aspect negatively.

We can examine other side-effects by checking if there is any other action a that occurs at step 1. Since the functional aspect is no longer satisfied, (2) will cause $\Omega \cup \Lambda \cup \mathcal{H}$ to yield $occurs(RejectFeed, 1)$, indicating that the court will reject the feed. Figure 4 illustrates the output of our system prototype, to demonstrate the potential of automated reasoning.

Mitigation. The last reasoning task we illustrate is aimed at determining how the effects of the attack can be mitigated. As before, let $\mathcal{H}$ be a set of occurrences of actions. We are interested in answering the query "which mitigation measure can restore γ?" where γ is a concern or the meta-aspect all.[1] Let us assume that the underlying logical language supports disjunctions $p \vee q$ (true if at least one of p, q is true). For sake of simplicity, let us focus on the case in which all mitigation actions are executed concurrently after the last action of $\mathcal{H}$. Let $s^{\#}$ denote the corresponding step. The query, $\mathcal{Q}$, is encoded by a statement $occurs(a, s^{\#}) \vee \neg occurs(a, s^{\#})$ for every action a that one is interested in allowing. The question is answered by finding the set of actions a such that $\Omega \cup \Lambda \cup \mathcal{H} \cup \mathcal{Q} \vdash \{holds(sat(\gamma), s^{\#}+1), occurs(a_1, s^{\#}), \ldots, occurs(a_k, s^{\#})\}$. In the BCAM use case, one can check that the mitigation action returned by this process is $MakeFalse(cam[basicOne])$, intuitively indicating that the basic camera should be replaced by the advanced camera in order to compensate for the fact that the cyberattack is forcing the CPS to record at 50 fps.

In the case of underlying languages that allow one to assign weights to statements and to require that the weight of satisfied statements be minimized, one can also use our approach to find optimal solutions. For instance, one might ask *"which mitigation measures can restore γ and involve the smallest number of actions?"*. The query can be encoded by extending $\mathcal{Q}$ so that it associates a weight of 1 to the statements of the form $occurs(a, s^{\#})$ and a weight of 0 to all other statements.

To illustrate the reasoning task, consider a variant of the BCAM use case, in which a SAM affected by the cyber-attack can be patched (action *Patch*) to force it to request a 25 fps recording rate. Let Ω and Λ be modified accordingly. Then, $\Omega \cup \Lambda \cup \mathcal{H} \cup \mathcal{Q}$ yields the two solutions $occurs(MakeFalse(cam[basicOne], s^{\#})$ and $occurs(Patch, s^{\#})$. While, in principle, another possible mitigation entails *both* replacing the basic camera *and* patching the SAM, it is ruled out because it has a non-minimal weight.

[1]For illustration purposes, we focus on after-the-fact mitigation. The same approach can cover preventive measures as well.

VI. Conclusion

In this paper, we used a simple example to illustrate how an extensive static model, such as the NIST CPS Framework, can be enriched by creating an ontology equivalent and developing reasoning capabilities in addition to the native capabilities of the CPS Framework. The availability of the CPS Framework allowed us to speed up the development of an ontology for a subset of the framework. The ontology permitted us to demonstrate the ability to gain additional insights into a use case through reasoning. Although the use case illustrates one complex activity associated with data-driven and connected Smart Cities, it highlights the enhanced ability to perform sophisticated analysis, such as determining indirect consequences of a cyber attack and of the use of a certain type of equipment. This work focused on trustworthiness, but the model contains sufficient complexity to demonstrate the capabilities of the approach and its scalability to a full CPS Reference Framework and to complex situations within the smart-city infrastructure. Our experiment includes complex cyber-physical considerations such as Transduction and Influence. Our work illustrates how an ontology-based methodology can aid engineers, operators, and city leaders in identifying and resolving important issues for the conceptualization, realization (including operation), and assurance of CPS that support Smart City infrastructures.

References

[1] B. Smith and C. Welty, "Ontology: Towards a New Synthesis," *Formal Ontology in Information Systems*, vol. 10, no. 3, pp. iii–x, 2001.

[2] E. Griffor, C. Greer, D. Wollman, and M. Burns, "Framework for Cyber-Physical Systems: Volume 1, Overview," National Institute of Standards and Technology, Tech. Rep. NIST-SP-1500-201, Jun 2017.

[3] P. Brizzi, D. Bonino, A. Musetti, A. K. E. Patti, and M. Axling, "Towards an ontology driven approach for systems interoperability and energy management in the smart city," in *Computer and Energy Science (SpliTech), Int'l Multidisc. Conf.* IEEE, July 2016, pp. 1–7.

[4] P. Bellini, I. Bruno, A. Cavaliere, D. Cenni, M. DiClaudio, G. Martelli, N. Rauch, and et al., "Km4City: Smart City ontology and tools for city knowledge exploitation," in *European Data Forum*, 2015.

[5] N. Komninos, C. Bratsas, C. Kakderi, and P. Tsarchopoulos, "Smart city ontologies: Improving the effectiveness of smart city application," *Journal of Smart Cities*, pp. 31–46, 2016.

[6] J. S. Luciano, D. S. Sayogo, W. Ran, N. Depaula, H. Jarman, L. Luna-Reyes, D. F. Andersen, and et al., *Private Data and Public Value*. Springer, 2016, ch. Using Ontologies to Develop and Test a Certification and Inspection Data Infrastructure Building Block, pp. 89–107.

[7] C. Porcel, C. Martinez-Cruz, J. Bernab-Moreno, A. Tejeda-Lorente, and E. Herrera-Viedma, "Integrating ontologies and fuzzy logic to represent user-trustworthiness in recommender systems," *Procedia Computer Science*, vol. 55, pp. 603–612, 2015.

[8] M. Balduccini, E. Griffor, M. Huth, C. Vishik, M. Burns, and D. A. Wollman, "Ontology-based reasoning about the trustworthiness of cyber-physical systems," *CoRR*, vol. abs/1803.07438, 2018.

[9] M. Gelfond and V. Lifschitz, "Action Languages," *Electronic Transactions on AI*, vol. 3, no. 16, 1998.