Peer-reviewed version available at *Sensors* **2017**, *17*, 1644; doi:10.3390/s17071644

Research Paper

Minding the Cyber-Physical Gap: Model-Based Analysis and Mitigation of Systemic Perception-Induced Failure

Yaniv Mordecai [1,3,*] **and Dov Dori** [1,2]

[1] Technion – Israel Institute of Technology, Haifa, Israel
[2] Massachusetts Institute of Technology, Cambridge MA, USA
[3] Motorola Solutions Israel, Airport City, Israel

* Correspondence: yanivmor@technion.ac.il; dori@ie.technion.ac.il;

Abstract: The cyber-physical gap (CPG) is the difference between the 'real' state of the world and the way the system perceives it. This discrepancy often stems from the limitations of sensing and data collection technologies and capabilities, and is an inevitable issue in any cyber-physical system (CPS). Ignoring or misrepresenting such limitations during system modeling, specification, design, and analysis can potentially result in systemic misconceptions, disrupted functionality and performance, system failure, severe damage, and potential detrimental impacts on the system and its environment. We propose CPG-Aware Modeling & Engineering (CPGAME), a conceptual model-based approach for capturing, explaining, and mitigating the CPG, on top of and in sync with the conventional system model, and as an inherent systems engineering activity. This approach enhances the systems engineer's ability to cope with CPGs, mitigate them by design, and prevent erroneous decisions, actions, and hazardous implications. CPGAME is a generic, conceptual approach, specified and demonstrated with Object Process Methodology (OPM). OPM is a holistic conceptual modeling paradigm for multidisciplinary, complex, dynamic systems, which is also ISO-19450. We analyze the 1979 Three Miles Island 2 nuclear accident as a prime example of the disastrous consequences of unmitigated CPGs in complex systems.

Keywords: Conceptual Modeling, Cyber-Physical Systems, Cyber-Physical Gap, Object-Process Methodology, Model-Based Systems Engineering, Three Mile Island 2 Accident.

List of Acronyms

CPD	Cyber-Physical Duality
CPG	Cyber-Physical Gap
CPGAME	Cyber-Physical Gap-Aware Modeling and Engineering
CPS	Cyber-Physical System(s)
MBSE	Model-Based Systems Engineering
OPD	Object-Process Diagram
OPL	Object-Process Language
OPM	Object-Process Methodology
SysML	Systems Modeling Language
TMI	Three-Mile Island
UML	Unified Modeling Language

1. Introduction

Complex *cyber-physical systems* (CPS) concurrently reside and act in the cyber and physical domains. Such systems mandate awareness, perception, and conception capabilities and technologies. Modern systems and solutions have become ever more cyber-physical, sentient, aware, and intelligent, as they harness the power of affordable sensors, endpoint computing, cloud-based knowledge and artificial intelligence. These facilitate the

capability to monitor, control, and engage actuators and physical agents, such as the human body, manned and self-driving cars, household robotics, smart homes, smart facilities and industries, and smart cities.

CPSs consist of a cyber-based segment and a physical segment [1,2]. The cyber segment is designed to manage, monitor, and control the physical segment. The physical segment senses, generates and sends stimuli to the cyber segment, which, in turn, uses them to evaluate the state of the system's physical segment and, not less importantly, of the environment. Furthermore, the cyber segment is designed to engage the physical segment, to actuate physical components, or to generate some physical impact on the environment.

CPSs still inherently face functional and technical challenges that stem from system-environment perception discrepancy – the system's or subsystem's inability to completely monitor, understand, and control its environment or even peer systems. Due to the availability and cohesion of technologies, these challenges become ever more present in common applications in commerce, services, and industry, as well as in everyday life. The cyber segment cannot capture all the aspects related to the state of the environment and the physical segment it controls. It cannot know the state of all the physical elements at any point in space at any given time. The result is a potential mismatch between the perceived or conceived state of the physical segment by the cyber segment and the actual state of the physical world—the environment or the physical segment.

The cyber-physical duality (CPD) is the notion that cyber-physical entities exist dually and concurrently as both physical embodiments and their informatical representations. Cyber-based agents that interact with external entities must therefore maintain an internal model of those entities so they can facilitate, manage, and control the interaction. For the interaction to succeed, it must use reliable, updated information about the entity's state and attribute values.

The *cyber-physical gap* (CPG) is the difference between the 'real' state of the world and the entities within it on the one hand, and the way the systems or subsystems perceive it on the other hand. The CPG is a disruption or disturbance to system's dynamics, behavior, functionality, performance, or output because of failure to correctly align the system's perception of the state of the environment (or the system's physical segment) with the corresponding actual state. The term CPG has emerged from the long-known distinction between the 'physical world' and 'virtual world' [3–6]. This discrepancy stems from technological, cognitive, and engineering inhibitors. The technological limitations of sensing, monitoring, data collection, data processing, inference, and automation technologies and capabilities, in machines as in humans, create the problem to begin with. Insufficient awareness of the problem by systems and disciplinary engineers alike, resulting in failure to acknowledge, foresee, capture, understand, explain, and mitigate these functional or technical inhibitors, is a key factor in their appearance and impact on the system and environment. Substantially, the inability to model, analyze, and design the system for coping with the CPG [7,8], reduces engineering competency and quality during the critical systems engineering phases, such as requirements engineering, system architecting, functional decomposition, risk analysis, technology selection, subsystem specification, and detailed design [9].

The combination of technological, cognitive, and engineering limitations has provably led to dire consequences and catastrophes, especially when CPG was involved – even if not exactly described as such [10,11]. The current state-of-affairs has caused and will continue to cause regular errors and failures in systems, but also catastrophes such as the 1979 Three Mile Island Nuclear Meltdown [12], and the 2014 Malaysia Airlines MH370 Disappearance [13].

There is a need to form an ontology of the CPG, to clarify its importance and ramifications to cyber-physical systems engineers, and to support CPG mitigation as part of ongoing systems specification and design. Moreover, there is a need for monitoring, mitigation, and control mechanisms and specification patterns that can be incorporated into cyber-physical systems architecture, and reduce the risks that CPGs pose to the system or the environment. CPG considerations must therefore be intertwined into normative systems thinking, modeling and engineering.

Conceptual models of phenomena and systems play a central role in science, engineering, and operations due to their descriptive and prescriptive values [14]. A model is a knowledge-base that captures and provides knowledge about a domain or a system-of-interest and information about attribute values or object states of generic types and specific instances. System architects, designers, developers, and operators use models to understand, specify, and communicate the system's function and architecture (structure-behavior combination) throughout its lifecycle. Modeling importance and significance is further emphasized under high or extreme variability, complexity, and risk [15].

Peer-reviewed version available at Sensors 2017, 17, 1644; doi:10.3390/s17071644

Technological constraints should be well-captured and well-handled as part of system modeling, specification, design, and analysis. Overly simplified or uninformed system models, which neglect or overlook perception factors and impacts of the CPG, result in incompetent systems. An unmitigated CPG can lead to the system's misconception of its environment, disrupted functionality and performance, risk or failure mode realization, emergency, and ultimately systemic failure, which results in anomalous behavior in the better case, and severe damage or even catastrophe in the worst case.

The primary contributions of this paper are hence: (i) a definition and formulation of the CPG concept, (ii) introduction of a formal, model-based, simple, tested, and verified ontological approach for capturing, considering, and controlling the CPG in complex cyber-physical systems, and (iii) a demonstration of this concept in a real-life case, with dire consequences, and a demonstration of the value that CPG-aware modeling and analysis could or would provide in cases of similar nature.

We propose CPG-Aware Modeling and Engineering (CPGAME[1]) – a conceptual modeling approach that provides for capturing, explaining, and mitigating the CPG in complex systems. We argue that such a modeling framework would be most useful and usable as an extension to a common model-based systems engineering (MBSE) framework, rather than as a stand-alone modeling paradigm. CPGAME advocates a holistic systems engineering process, based on a rich conceptual model that covers functional and technical system aspects, which has the capability to describe CPG and potential CPG-induced disruptions to the system and its environment. Accordingly, CPGAME is based on Object-Process Methodology, OPM [16], a holistic conceptual modeling and simulation paradigm for multidisciplinary, complex, and dynamic systems and processes. OPM is ISO 19450 [17], and a state-of-the-art methodology and paradigm in both the conceptual modeling and MBSE domains [18–20].

A CPG-aware design may emerge during the modeling and engineering process without an officially-adopted CPG-aware approach. Rather, this may be a result of a lesson learnt from previous projects, a clearly-stated requirement, a compelling necessity, an iterative design insight, or a demonstrated failure in early prototypes. Our purpose in this paper is to make the evolutionary conceptual modeling, design, and engineering process more conscious, effective, expressive, and informative by accounting for the CPG problem and modeling elements to solve it.

The rest of this paper is organized as follows: Chapter 2 includes a literature review regarding the CPG and various synonymous and analogous concepts, and the state-of-the-art in CPG modeling and analysis. It also includes a brief introduction of OPM, our underlying modeling paradigm. Chapter 3 includes a specification of the CPGAME approach, including an enhanced modeling process and a modeling pattern that covers the various aspects related to the CPG and its mitigation. In Chapter 4, we demonstrate CPGAME on the Three Mile Island nuclear reactor partial meltdown accident of 1979, known as "TMI2", which constitutes a benchmark case in system safety and risk analysis, but also a compelling example of a CPG-induced catastrophe. We discuss the results, conclude this paper, and propose directions for future research in Chapter 5.

2. Related Work

2.1. Cyber-Physical Gap: The Discrepancy between Reality and Its Conception

CPG has been a primary concern in cybernetics, informatics, epistemic logic and knowledge representation for a long-time even if not named this way before. Its role in natural, societal, and technical processes is acute [3,5,21–23]. Several studies on real-world event and information detection, representation, and response are available in the literature, with applications in system safety [10,24,25], cyber security [26–28], and counter-terrorism [29,30]. The precise term Cyber-Physical Gap has recently been mentioned in the context of the Internet of Things [31], as web browsers' inability to interact with device-integrated sensors and actuators, and the resulting limited context-awareness. Nevertheless, we found this usage of the term quite narrowing, relative to the potential scope it could apply to.

CPG results at least in part from the limited or missing ability of system models to precisely capture the intricacies of the systems they specify. Different models present different views related to various aspects of the system, such as CPG. Models vary in quality and fidelity, and partially-understood systems suffer from some discrepancy between them and the models that describe them. This discrepancy is the basis for the potential

[1] Pronounced "C.P.Game".

Peer-reviewed version available at Sensors 2017, 17, 1644; doi:10.3390/s17071644

approximation-complexity tradeoff and resulting multiplicity of and variability among several models of the same problem [32,33]. The hierarchical detail decomposition approach to complexity management is the strategy OPM employs to resolve the clarity-completeness tradeoff—the need to provide a clearly understandable system specification on one hand and a complete one on the other hand [16].

The literature on CPG-related aspects does not provide a holistic model-based approach to incorporate CPG notions such as detection, classification, representation, prevention, and mitigation into the conceptual model, architecture, design, and specification of the CPS. As long as this problem is treated without reference to or integration with the core system model, the ability to mitigate the CPG by design would be severely limited. Worse, considering CPGs as features of a black box CPS, i.e., without delving deep into the CPS's internal parts in order to understand, address, and resolve the root causes and impacts, transfers the responsibility to contain and mitigate adverse CPG-related impacts by-design from the CPS to the systems that interact with it and human users or operators.

Our study of CPG and CPG-aware systems modeling, analysis, and engineering included theoretical and fundamental principles of the physical-informatical essence duality (PIED) of cyber-physical entities, and PIED-aware conceptual modeling semantics [7] and integration of CPG considerations into automated decision-making [34] and systems-of-systems interoperability [35]. We also analyzed the occurrence of CPG in various cases, including the lost baggage problem [8] and cyber threat detection and response [36]. The CPG in air traffic control was also proposed as a major factor in the disappearance of Malaysia Airlines flight MH370 in the Indian Ocean in 2014 [8]. The need to provide a holistic modeling framework for defining and managing CPG has motivated the additional research that resulted in the present paper.

2.2. *Object-Process Methodology (OPM)*

Model-Based Systems Engineering, MBSE, is the application of formalized conceptual modeling to complex sociotechnical systems through their entire lifecycle. MBSE plays a key role in this context, and is gaining increasing momentum and adoption as the basis for systems engineering, with OPM being one of several leading, state-of-the-art methodologies for MBSE, and as a notable modeling language next to languages like the Unified Modeling Language (UML), UML's systems-oriented profile – SysML, Enhanced Functional Flow Block Diagrams (EFFBD), Mathwork's Simulink, Integrated Definition Framework (IDEF), and others [19,20,37].

OPM features a unique capability to capture functional, structural, and procedural aspects of any natural or artificial system in a unified manner. OPM is founded on the universal ontology principle, according to which two kinds of things—stateful objects and processes that happen to objects and transform them—along with relations among them constitute a necessary and sufficient set of elements for conceptually specifying systems and phenomena in any domain in the universe.

OPM models are bimodal—they are expressed by a graphical modality alongside a textual one. OPM consists of a compact set of graphical-textual elements – **Things** and **Links** [2]. **Process** (ellipse) and **Object** (rectangle) are **Things**. Objects can be stateful, i.e., have states. Things inherently exhibit **Essence**, which can be **physical** or **informatical**, and an **Affiliation** attribute, which can be **systemic** or **environmental**. Links express various relations between **Things**. **Structural Links** support the modeling of static system aspects. **Procedural Links** express procedural relations, control, and causalities.

The graphical view consists of a coherent interconnected set of hierarchically organized Object-Process Diagrams (OPDs). Each OPD extends its ancestor OPD by adding new details, providing for inherent complexity management and reduction. The same compact set of elements (things and relations among them) is used at any level in the OPD hierarchy, so all the OPDs in any OPM model are self-similar, regardless of their level of detail. The hierarchically-organized OPDs are derived using several refinement-abstraction mechanisms: (i) unfolding and folding of structural hierarchies of things, (ii) zooming into or out of inner details of things, (iii) state expressing (showing) and suppressing (hiding), and (iv) view creating for expressing additional information. Every MF needs to appear at least once in some OPD in order for it to be valid throughout the model.

Object-Process Language (OPL) is a structured textual specification English-like language. Each OPD is accompanied by an OPL paragraph, in which each graphical OPD construct specifies some model fact (MF) that is also expressed by a semantically equivalent textual OPL sentence.

[2] **Bold Arial** font represents names of things (objects or processes) or states in the OPM model. **Green** names indicate objects. **Blue** indicates processes. **Gold** indicates object states.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2017 doi:10.20944/preprints201706.0113.v1

Peer-reviewed version available at Sensors 2017, 17, 1644; doi:10.3390/s17071644

OPM is ISO 19450, [17] and it features a freely available CASE tool: OPCAT3 [38], a graphical application for model creating, managing, and sharing. OPCAT supports most OPM concepts and immediately validates model edits according to OPM grammar. OPCAT automatically generates OPL text in response to graphical model edits and provides whole-model or diagram-level textual OPL specifications. OPCAT has a built-in qualitative simulation engine, which supports model validation, verification, and testing, and is especially useful in visualizing the execution of complex interactions.

OPM's simplicity on the one hand and its expressive power on the other hand enable clear and concise modeling and architecting of nominalism in conjunction with disruption. This feature is especially significant for modeling CPG thanks to the ability to model physical things alongside informatical ones, as we elaborate next.

3. CPG-Aware Modeling & Engineering – CPGAME

3.1. CPG-Awareness

We begin with the definition of the system's knowledge base (KB): a (mostly static) collection of knowledge classifications and abstractions the system is equipped with. This includes what the system's cyber-segment-based software agent needs to know about the physical environment in general and physical entities that it has to handle or tackle. Furthermore, the system's information base (IB) includes what the system dynamically knows during its real-time operation about that physical environment, based on information it has gathered or received. Although both the KB and IB of the system are stored and managed in a database (DB), the CPG-aware approach emphasizes this knowledge-information distinction, which is analogous to the distinction between the conceptual model level (knowledge about the relevant world) and the runtime level (ideally near real-time information about the relevant known world).

We distinguish five levels of knowledge about and conception of a physical entity: (a) kind awareness, (b) instance perception, (c) feature existence awareness, (d) state-space awareness, and (e) state perception. Table 1 defines these levels and exemplifies how a carwash control system would refer to each level. The kind, property, and state-space awareness levels belong to the system's KB. These generic awareness levels are intertwined with the perception levels, which belong to the IB.

Table 1. Levels of Systemic Conception of Reality

Conception Level	Definition	Example
Kind Awareness	knowledge and definition of the entity kind	Carwash system is aware of the possible existence of a class of trucks in addition to the class of cars.
Instance Perception	awareness of existence of a specific instance of a known entity kind	Carwash system detects the presence of a new truck in the carwash tunnel.
Feature Existence Awareness	existence of a significant feature—a part, an attribute, or an operation of the entity	Carwash system knows the brand, length, width, height, and GPS antenna location of the detected truck.
State-Space Awareness	the possible set of states or range of values that an entity or any of its attributes can assume	Carwash system knows all potential car trunk configurations, including hatchback, sedan, station wagon, and cabriolet.
State or value Perception	the specific state of the entity or the value of any one of its attributes at any relevant point in time	Carwash system knows that the car currently in tunnel is a cabriolet.

With respect to the five awareness and perception levels, we observe three classes of CPG, which apply at the various conception levels, and represent three phases of CPG-related failure. Each class has two types, which represent the causes for the failure. The three classes and their six respective types are summarized in Table 2.

[3] Downloadable for free from http://esml.iem.technion.ac.il/

Peer-reviewed version available at Sensors 2017, 17, 1644; doi:10.3390/s17071644

Table 2. Classes and Types of CPG

Class	Type
Class A: failure to detect or identify a detectable entity	Type A1: detection mechanism problem or lack of a pattern that resembles the entity
	Type A2: acquisition mechanism problem or inability to perform entity pattern-matching
Class B: failure to generate a coherent representation of an acquired entity	Type B1: problem with representation of the acquired entity's state
	Type B2: problem with representation of the acquired entity's behavior
Class C: failure to generate a coherent interaction with an acquired entity	Type C1: problem with generating the intended interaction
	Type C2: problem with generating the intended result of an interaction

Intelligent systems might be able to detect and infer the state of external entities based on intrinsic and exogenous sources of information. The internal representation of the external entity by the system may therefore be fuzzy, consisting of several possibilities, each with its probability or likelihood [39]. When several sources of information are available, they can yield inconsistent or even contradicting results. For example, in the carwash example in Table 1, two sensors can indicate different trunk configurations: one can sense "hatchback" while the other—"sedan". The system must therefore include a dedicated mechanism for fusion, reconciliation, conflict-resolution, selection, or determination over the possible state-set of the uncertain entity or one of its properties. One solution can be voting among three sensors. Alternatively, in some cases, the result can be fuzzy: the entity may be defined as existent with probability p or non-existent with probability 1–p; the entity's state may become a probability distribution over a state-set [40]. The performance of the decision mechanism that combines several or multiple information sources can improve over time if machine learning is applied [41]. Conflict resolution may require human intervention in order to help the system make its final decision or to enforce a judicious decision on the system.

3.2. CPG-Aware Modeling with OPM

The transition from a nominal, or naïve model, towards a CPG-aware model is a gradual process of evolutionary conceptual modeling. During this process, the system architect must identify cases of CPG in the system-environment interaction and incorporate into the model the notion of CPG along with the system's response to that gap. In this section, we gradually apply the principles and guidelines of CPGAME in order to evolve the model from being naïve to CPG-aware. We use OPM as our conceptual and ontological modeling framework, and define a generic pattern model that:

a. expresses the CPD and the various CPG forms on top of the nominal system model,

b. specifies the monitoring, control, and mitigation mechanisms to be incorporated into the CPS in order to cope with and bridge the CPG, and

c. represent the various precarious situations caused by the CPG and how they prove to be resolved by executing the model via animated simulation.

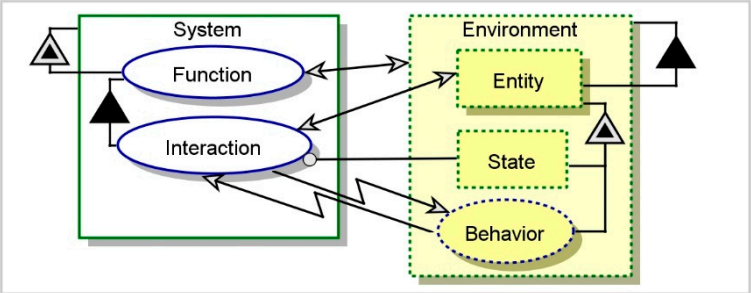
We begin with a naïve generic model of system-environment interaction. Figure 1 shows the *system diagram* (SD)—the top-level OPD—of an OPM model in both its graphical and textual modalities. This model shows an **Interaction** of the **System** with an **Entity** in the **Environment**, as part of the **Function** of the **System**. The **Interaction** is based on the actual **State** of the **Entity**. It invokes, the **Entity**'s **Behavior**, and is cyclically invoked thereby.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2017 doi:10.20944/preprints201706.0113.v1

Peer-reviewed version available at Sensors 2017, 17, 1644; doi:10.3390/s17071644

The primary reason for referring to the model in Figure 1 as *naïve* is that it implicitly assumes that the system can interact with the environment based on the environment's actual state. In reality, however, the interaction is based on the internal representation of the environment as sensed and interpreted by the system, giving rise to a potential CPG. To account for CPG, the model has to include an explicit definition of that internal representation of the environment and consider cases of misrecognizing elements in it.

Figure 2 presents a revised SD, in which the CPG aspect is weaved into the model[4]. This enhanced model illustrates the interaction between the **System** and its **Environment** through the **CPG-Aware Interaction** process, which emerges from both the **Action** of the **System** and the environmental **Entity Behavior**. The **Action** generates a **Cyber-Physical Effect**, which triggers **Entity Behavior**. The **Entity** generates a physical **Footprint** of its presence, and a physical **Impact** on the **Environment**. The system must detect and understand both the **Footprint** of and the **Impact**. The **Action** of the **System** relies on the **Internal Representation**, which relates to the external **Entity**. This **Internal Representation** includes **Existence Representation**, **State Representation**, and **Behavior Representation**. The functionality for handling these representations is contained in the conceptual process **Cyber-Physical Gap Managing**.



ID	Statement
100	System is physical.
200	System exhibits Function.
210	Function is physical.
220	Function consists of Interaction.
221	Interaction is physical.
222	Interaction requires State.
223	Interaction affects Entity.
224	Interaction invokes Behavior.
230	Function affects Environment.
300	Environment is environmental and physical.
400	Environment consists of many Entities.
410	Entity is environmental and physical.
420	Entity exhibits State, as well as Behavior.
421	State is environmental.
422	Behavior is environmental and physical.
423	Behavior invokes Interaction.

Figure 1. System diagram of a generic naïve system-environment interaction

The CPG-aware model demonstrates a new *second-order modeling* pattern: the **Environment Representation** that the **System** holds is an informatical object inside the model that is also a model that is used by the system in run-time. OPM's universal ontology principle and metamodeling capability are significant enablers of this modeling approach; OPM provides the systems engineer with the means to model both the system as a whole and the internal model of the environment that the system maintains. The modeling language for both modeling layers is one and the same.

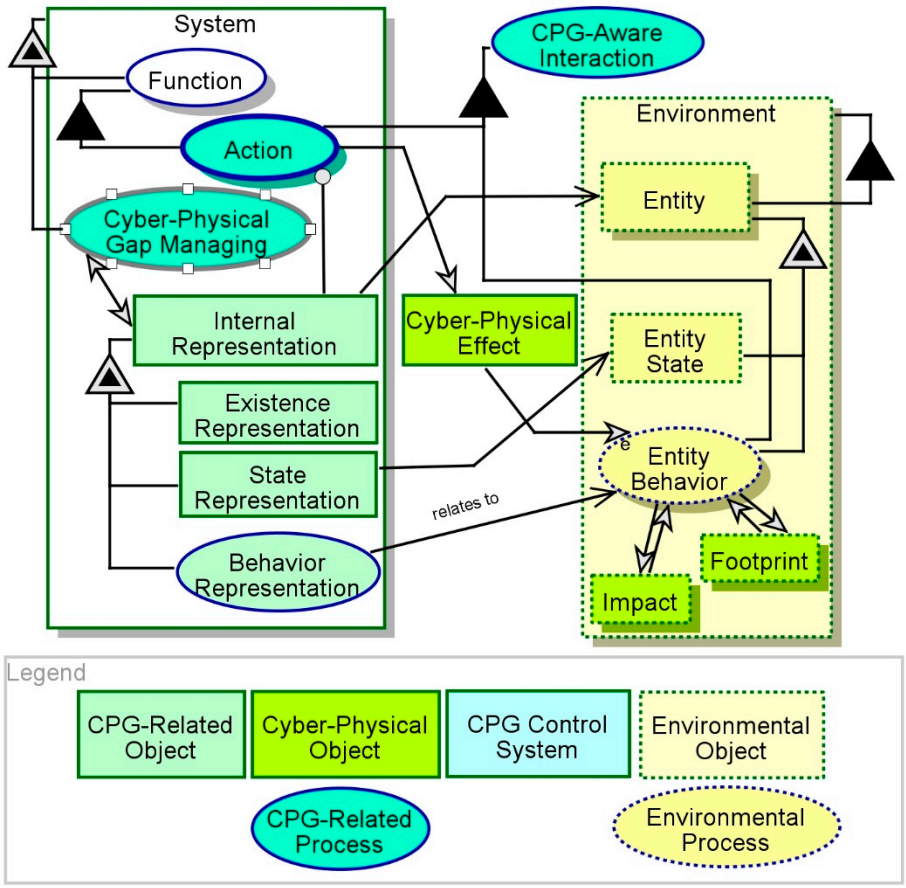
[4] The generic CPGAME model is available online at https://1drv.ms/f/s!AsN2SH2tvCOWjmu6bRvpHS1pyiWm. Usage is permitted under the terms of the CreativeCommons CC BY-NC 4 license.

In Figure 3, **Cyber-Physical Gap Managing** from Figure 2 is unfolded to reveal three subprocesses: **Entity Acquisition**, **Representation Management**, and **Interaction Analysis**. There is a clear reliance of these processes on the system-maintained **Internal Representation**, in addition to the entity's **Footprint** and **Impact**, in order to capture the external **Entity** or interact with it.

A CPG can emerge at any phase during the system operation. The system may fail to acquire an external entity even if the entity is present in the scene, and consequently disregard that entity. The system can misidentify or misrepresent an acquired entity, and therefore mishandle it. Finally, the actual outcome of an interaction with the entity may not match the outcome that the system expects. A CPG-aware system must implement acquisition, representation, and interaction to cope with these challenges and be resilient to the CPG's undesired results. In what follows, we specify the CPG-aware approach to these three critical functionalities.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2017 doi:10.20944/preprints201706.0113.v1

Peer-reviewed version available at Sensors 2017, 17, 1644; doi:10.3390/s17071644



ID	Statement
100	System is physical.
200	System exhibits Function and Cyber-Physical Gap Managing.
210	Function is physical.
220	Function consists of Action.
221	Action is physical.
222	Action requires Internal Representation.
223	Action yields Cyber-Physical Effect.
230	Cyber-Physical Gap Managing affects Internal Representation.
400	Environment is environmental and physical.
500	Environment consists of many Entities.
510	Entity is environmental and physical.
520	Entity exhibits Entity's State, as well as Entity's Behavior.
521	Entity's State is environmental.
522	Entity's Behavior is environmental and physical.
523	Entity's Behavior consumes Impact, Footprint, and Cyber-Physical Effect.
524	Entity's Behavior yields Impact and Footprint.
600	Internal Representation exhibits State Representation and Existence Representation, as well as Behavior Representation.
610	State Representation relates to Entity's State.
620	Behavior Representation relates to Entity's Behavior.
700	At least one Internal Representation relates to Entity.
800	Cyber-Physical Effect triggers Entity's Behavior.
900	Footprint is environmental and physical.
1000	Impact is environmental and physical.
1100	CPG-Aware Interaction consists of Action and Entity's Behavior.
1200	Environmental Process is environmental.

Figure 2. System diagram of a CPG-aware system. Highlighted lines are additions to the naïve model.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2017 doi:10.20944/preprints201706.0113.v1

Peer-reviewed version available at Sensors 2017, 17, 1644; doi:10.3390/s17071644

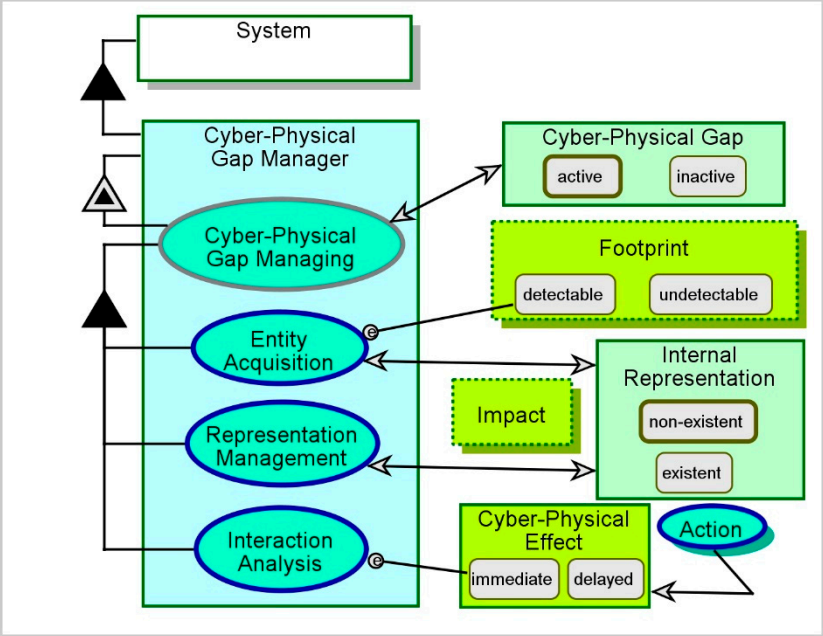


Figure 3. The Cyber-Physical Gap Managing process unfolded

3.3. *Entity Acquisition*

Entity acquisition is the system functionality for detecting entities in the environment. Acquisition must occur before the syste can interact with the entity. The assumption that the system is aware of any such entities by default is wrong and misleading. This assumption can hardly be valid at early design stages for static relations, such as fixed interfaces among fixed and reliably-connected subsystems, let alone more complicated situations. For a system to become resilient to subsystem connectivity issues, these considerations will eventually have to be incorporated into the design. As the design progresses, it will be wiser to assume that subsystems may be unaware of the existence and state of any of their peer subsystems, let alone external entities.

The issue of awareness and acquisition of external subsystems is also challenging in the context of agent-based systems. An agent in this context is any loosely-connected system or device, that has to connect to a central control system in order to commence and maintain some valuable interaction. Consider, for instance, a mobile device with a navigation application that connects to a central application server in order to obtain map and traffic updates, report its position and speed, send route requests, and receive route suggestions. For these interactions to take place, the central control system and each individual device first must become aware of its peers. Controlled, off-line binding may be useful when a manageable number of agents are deployed, but in general one must not assume that two endpoints are by default aware of each other such that interaction can take place with no prior check.

The third case for entity acquisition concerns transient entities with which the system may interact. Here are three examples: a) Baggage items pass through an airport baggage handling system, which must determine their destination according to the barcode that is taped to them. b) Airplanes travel through an air traffic control (ATC) area, and the ATC system must identify, track, and guide them. c) Customers walk into a shopping mall or bank branch and are interested in special offers, sales, or services. In all these cases, the system must include suitable means to identify and acquire transient entities with which it is required to interact.

Figure 4 is a model of a generic **Entity Acquisition**. The process begins when a **detectable Footprint** is generated due to the external, uncontrolled **Entity Behavior**. Even if the **Footprint** is **detectable**, the **Entity Detecting** phase (the first subprocess) may result in either **successful** or **unsuccessful Detection**. An **unsuccessful Detection** generates **CPG Type A1** – failure to detect a detectable impact, due to failure in the detection mechanism. A **successful Detection** leads to **Entity Matching**, if an **Internal Representation** is already **existent**. Otherwise, **Entity Acquiring** generates an **existent Internal Representation** in the first detection of the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2017 doi:10.20944/preprints201706.0113.v1

Peer-reviewed version available at Sensors 2017, 17, 1644; doi:10.3390/s17071644

entity. This can only be done if the **Entity Pattern** is **available** – i.e. if the **Footprint** of the external **Entity** is sufficiently similar to an existing pattern. Like **Detection**, **Acquisition** can also be either **successful** or **unsuccessful**. When **Acquisition** is successful, it leads to **Entity Registering**, which generates a **coherent Existence Representation** – at this point this is merely acknowledging the existence of an external entity whose **Footprint** matches some existing pattern. When **Acquisition** is **unsuccessful**, this results in **CPG Type A2** – failure to acquire a detected entity. **Acquisition Completing** invokes **Representation Management**, in which the details of the representation are treated. A learning system also updates the **Entity Pattern**, so that even if **CPG Type A1** is **active**, it might be neutralized the next time a **detectable Footprint** is successfully detected by the system.

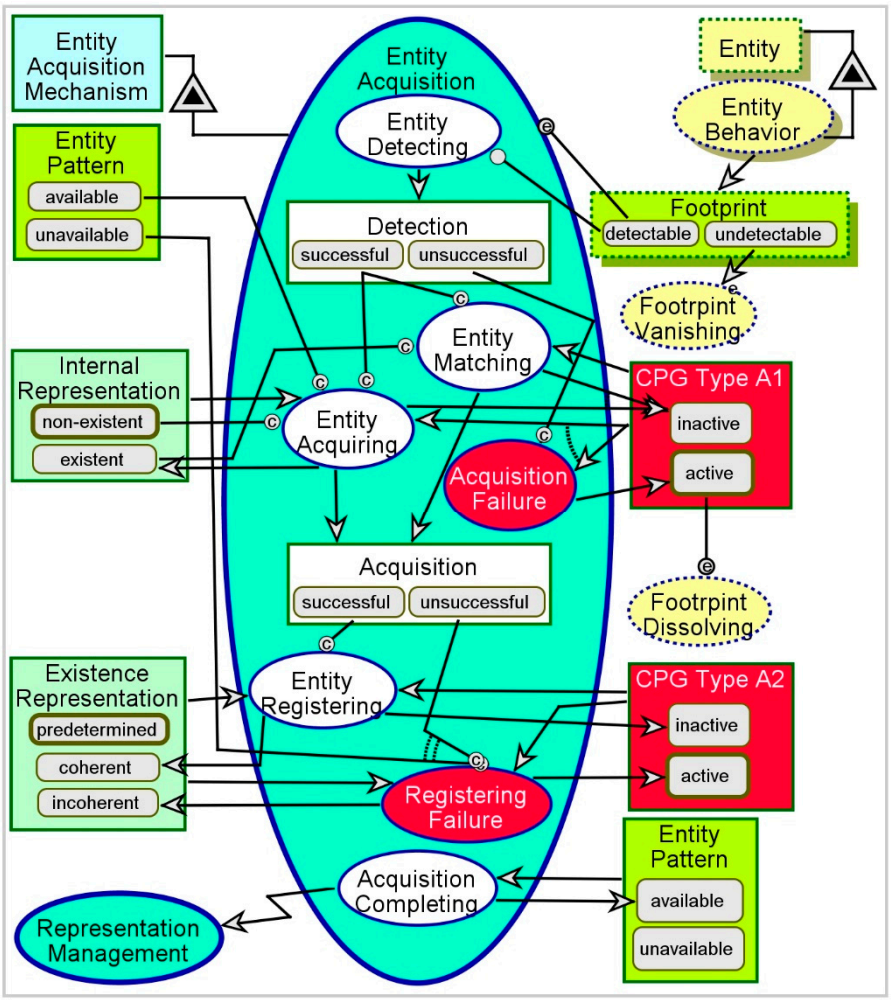


Figure 4. The Entity Acquisition process

To simulate **Entity Acquisition** using OPCAT in simulation mode, we begin with activating **Entity Behavior**. This will randomly generate either **detectable** of **undetectable Footprint**. An **undetectable Footprint** will trigger **Footprint Vanishing**, which will then consume **Footprint**. A **detectable Footprint** will initiate **Entity Acquisition**. **Entity Detecting** will randomly result in either **successful** or **unsuccessful Detection**. In the first time **Entity Acquiring** will occur, provided that: i) **Detection** is **successful**, ii) **Entity Pattern** is **available**, and iii) **Internal Representation** is **non-existent**. **If Internal Representation** is already **existent**, **Entity Matching** will occur instead. If **Detection** is **unsuccessful**, **Acquisition Failure** will occur, and the **Entity Acquisition** process will terminate. **Entity Acquiring** and **Entity Matching** both result in either **successful** or **unsuccessful Acquisition**. If **Acquisition** is **successful**, **Entity Registering** will occur. Otherwise, **Registering Failure** will occur. After each completion of **Entity Acquisition**, the process can be repeatedly simulated by re-activating **Entity Behavior**. Various combinations of output variables will be generated, including four possible combinations of **CPG Type A1** and **CPG Type A2**.

Peer-reviewed version available at Sensors 2017, 17, 1644; doi:10.3390/s17071644

3.4. *Representation Management*

Representation management allows the system to develop a sufficiently precise understanding of the acquired entity's state and behavior, and refine the internal representation so it can support system-entity interaction. The representation of the entity evolves and improves with each cycle of analysis of the entity's behavior, its footprint, and its observable attributes. Wrong interpretation of these properties of the entity may mislead the system. Fusion of multiple sensors or data sources, along with machine learning and proactive monitoring and data collection should be applied, in order to a) reduce the likelihood of inference error, and b) converge towards reliable and precise representations.

A model of the **Representation Management** process is shown in Figure 5. The process begins with an invocation by the previous process – **Entity Acquisition** – or by self-reactivation of the process by itself, using the **Representation Management Trigger**. First, **State Acquiring** occurs, and the values of state attributes of the entity are acquired according to the **State Attribute Set** stored as part of the **Entity Pattern**. Not all the attributes of the entity may be acquired, and those state attributes, which are critical or necessary for successful interaction, must be defined in the system's KB so that it may track them. It is necessary that all the **State Attributes** be acquired correctly for the **State Acquisition Result** to be a **desired** one, which qualifies the **State Representation** as **coherent**. Otherwise, it is an **undesired** result, and the **State Representation** remains **incoherent** as initially assumed. This reasoning is projected as the **CPG Type B1** – failure to acquire the state of an acquired entity.

Transient entities may be friendly, passive, or adversary. A friendly entity, such as a mobile device or remotely operated semi-autonomous vehicle, may communicate with the central control and provide details on its whereabouts and intentions. A passive entity, such as a barcode- or RFID-tagged baggage, cargo, or merchandise, may only react to attempts to communicate with it and respond or reply as needed. An adversary entity, such as an intruder, malicious software, hostile aircraft, or attacking ballistic missile, may try to evade the system, refuse to communicate, mislead and confuse it, and even disrupt or attack the system directly. In many cases, the system must include suitable means to determine the friendliness classification for the transient entities that it acquires. This is especially critical in environments that host a mixture of friendly and adversary entities, such as a battlefield, air traffic corridors, or computer networks. The **Friendliness Representing** process determines whether the acquired entity's **Friendliness** is **positive**, **neutral**, or **negative**. This optional process must be implemented in cases like the abovementioned ones. The system must hold some **Friendliness-Indicative Attribute Set** as part of the **State Attribute Set** to determine **Friendliness**. Designers must be aware of the possibility that such attributes may be exploited by an adversary to mislead the system to consider an entity friendly or neutral. Hence, friendliness evaluation may also be susceptible to a special case of **CPG Type B1**.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2017 doi:10.20944/preprints201706.0113.v1

Peer-reviewed version available at Sensors 2017, 17, 1644; doi:10.3390/s17071644

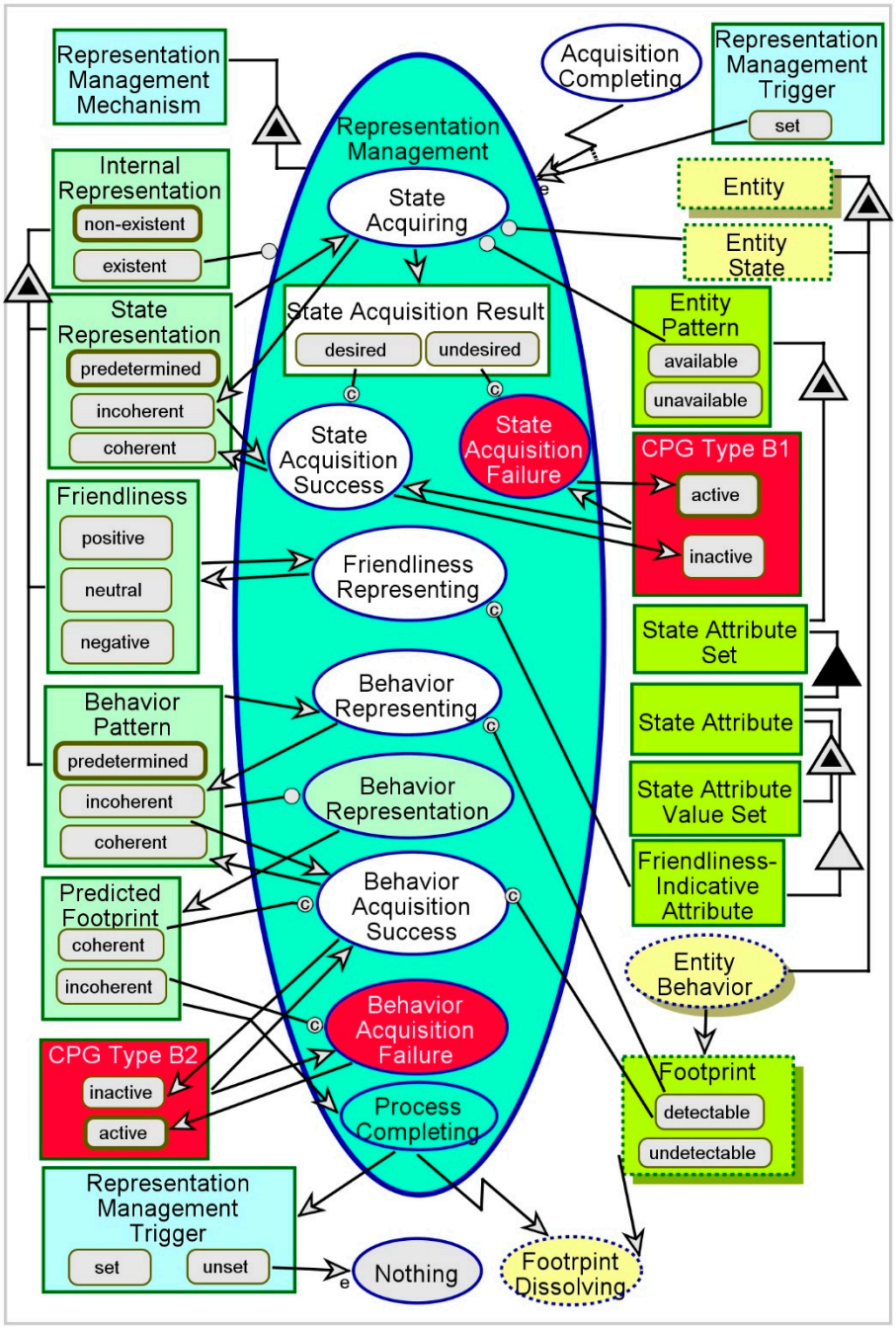


Figure 5. The Representation Management process

The third part of **Representation Management** includes the representation of the entity's behavior. The **Entity Behavior** is inferred from its **detectable Footprint** – the same **Footprint** that triggered the **Entity Acquisition** process. The system derives a **Behavior Pattern**, which is injected into the **Behavior Representation** model that the system holds as part of the **Internal Representation**. **Behavior Representation** is executed by the system and a **Predicted Footprint** is generated. If the **Predicted Footprint** is **coherent**, i.e., consistent with the **Entity**'s **detectable Footprint**, this means that the inferred **Behavior Pattern** is also **coherent**. Otherwise, this means that the **Behavior Pattern** is **incoherent** – a situation defined as **CPG Type B2**. When **Representation Management Process Completing** occurs, it determines whether another **Representation Management** iteration is needed to refine the results, and generates the **Representation Management Trigger**, which triggers another iteration if it is **set**. Considerations for additional iterations are however beyond the scope of the current paper.

Simulating **Representation Management** using OPCAT must follow at least one successful completion of **Entity Acquisition**, so that the **Internal Representation** will be created. As explained, activating **Entity Behavior** and generating a **detectable Footprint** will allow **State Acquiring**, **Friendliness Representing**, and **Behavior**

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2017 doi:10.20944/preprints201706.0113.v1

Peer-reviewed version available at Sensors 2017, 17, 1644; doi:10.3390/s17071644

Representing. The model is designed to generate random results for internal control/decision variables such as **State Acquisition Result**, **Predicted Footprint**, and **Representation Management Trigger**. Consequently, any of the four possible combinations of **CPG Type B1** and **CPG Type B2** may emerge.

3.5. Action and Interaction

Cyber-physical systems function to obtain their goal or serve their purpose by interacting with entities within the environment. Therefore, we first specify the actions and interactions from a naïve, value-providing vantage point. Nevertheless, applying the CPGAME approach means that any interaction is based on an internal representation rather than on an external state that is assumed to be known to the system. A mismatch between the internal representation and the actual manifestation—a CPG—may result in incoherent system behavior or entity's response. The model must therefore specify the disrupted, CPG-aware case next to the nominal one, in order to clarify the ramifications of incoherent action and interaction, providing the following benefits:

a. Understanding of the implications of incoherent actions,

b. Ability to simulate, map, and analyze the possible paths, especially those leading to failure,

c. Compliance with regulations or safety requirements, and

d. Incorporation of engineering or operational risk mitigation mechanisms.

Figure 6 illustrates the system's **Action**, which results in a **Cyber-Physical Effect**, and includes **Nominal Action** and **Disrupted Action**. **Disrupted Action** may occur instead of or in addition to **Nominal Action**. **Action** occurs only if the system has an **existent Internal Representation**, since otherwise it has nothing to refer to, even if the external **Entity** is present and engaging the system. **Nominal Action** occurs if both the **Existence Representation** and the **State Representation** are **coherent**. Otherwise, **Disrupted Action** occurs. Obviously, **Nominal Action** ends in a **coherent Action Outcome**. **Disrupted Action** can result in either a **coherent** or an **incoherent Action Outcome**. The **Cyber-Physical Effect** triggers and affects the external **Entity Behavior**. A **coherent Cyber-Physical Effect** results in nominal, coherent **Entity Behavior**. An **incoherent Cyber-Physical Effect** may be met with coherent **Entity Behavior**, depending on the **Entity**'s robustness, friendliness, and intelligence, which are not specified in this diagram. Regardless of the designer's assumptions, the system must analyze the results to determine whether the interaction was successful.

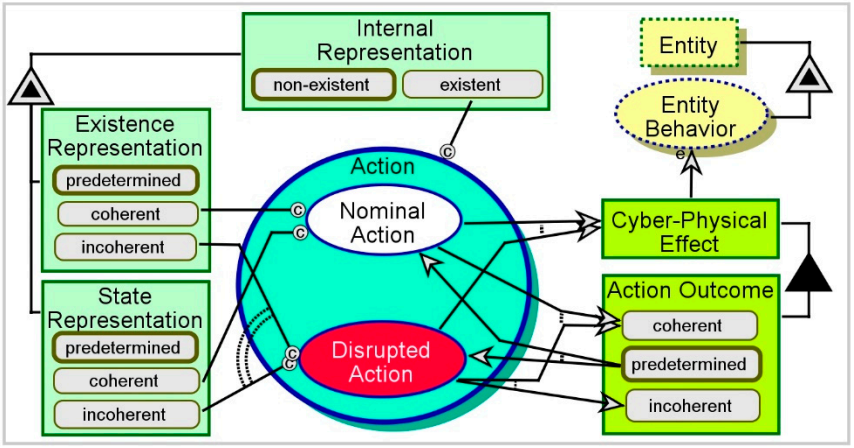


Figure 6. The system's Nominal Action or Disrupted Action

3.6. *Interaction Analysis*

The third aspect of CPG-awareness and third phase of CPGAME is **Interaction Analysis**. In the first two phases – **Entity Acquisition** and **Representation Management** – the system had to correctly acquire information about the **Entity**. When the CPS exhibits the functionality of interaction with the external entity, it must analyze this interaction and determine whether it has occurred the way it was intended, and whether its results match the intended or expected results. This analysis can be conducted in real-time or immediately thereafter, depending on the system, interaction, and result criticality.

If the internal representation is incoherent, i.e., if any of **CPG Types A1**, **CPG Types A2**, **CPG Types B1**, or **CPG Types B2** is **active**, undesired interaction pattern or undesired results can occur. Failure to obtain the intended or expected interaction is defined as **CPG Type C1**, while failure to generate the expected result (regardless of whether the interaction was as planned or not) is define as **CPG Type C2**.

A CPG-aware model has to account at least for the way the system uses the internal representation that it holds – rather than the actual state of the entity being represented – to interact with the external entity. The model has to account for the impact of incoherent interaction or its incoherent results, while taking into account the possible latency of the occurrence or detection of the impact of interaction. Finally, the model has to cover techniques to identify and mitigate **CPG Type C1** and **CPG Type C2**.

Acquiring the interaction's course and outcome is similar to the acquisition of **Entity behavior** and the **Entity**'s **State**, as we can see in Figure 7. As shown, this procedure is triggered by the **Cyber-Physical Effect**, which represents the impact of the system's **Action** on the **Entity**. An **immediate Cyber-Physical Effect** triggers **Interaction Analysis**, while a **delayed** effect means waiting until some external event changes it to **immediate**. There are two main parts in this procedure – first, determining whether the interaction itself is as intended, as inferred from internally simulating the **Behavior Representation**; second, determining whether the result of the interaction is as intended, as indicated by the **Impact** of the **Entity** on the **Environment**. The outcome of the first part is a determination of **CPG Type C1** – failure to conduct the interaction as expected. The outcome of the second part is a determination of **CPG Type C2** – failure to obtain the desired result or intended impact. The reason for incoherent interaction is most likely incoherent representation of the entity, since the action taken by the system as part of interaction with the entity is based on the internal representation.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2017 doi:10.20944/preprints201706.0113.v1

Peer-reviewed version available at Sensors 2017, 17, 1644; doi:10.3390/s17071644

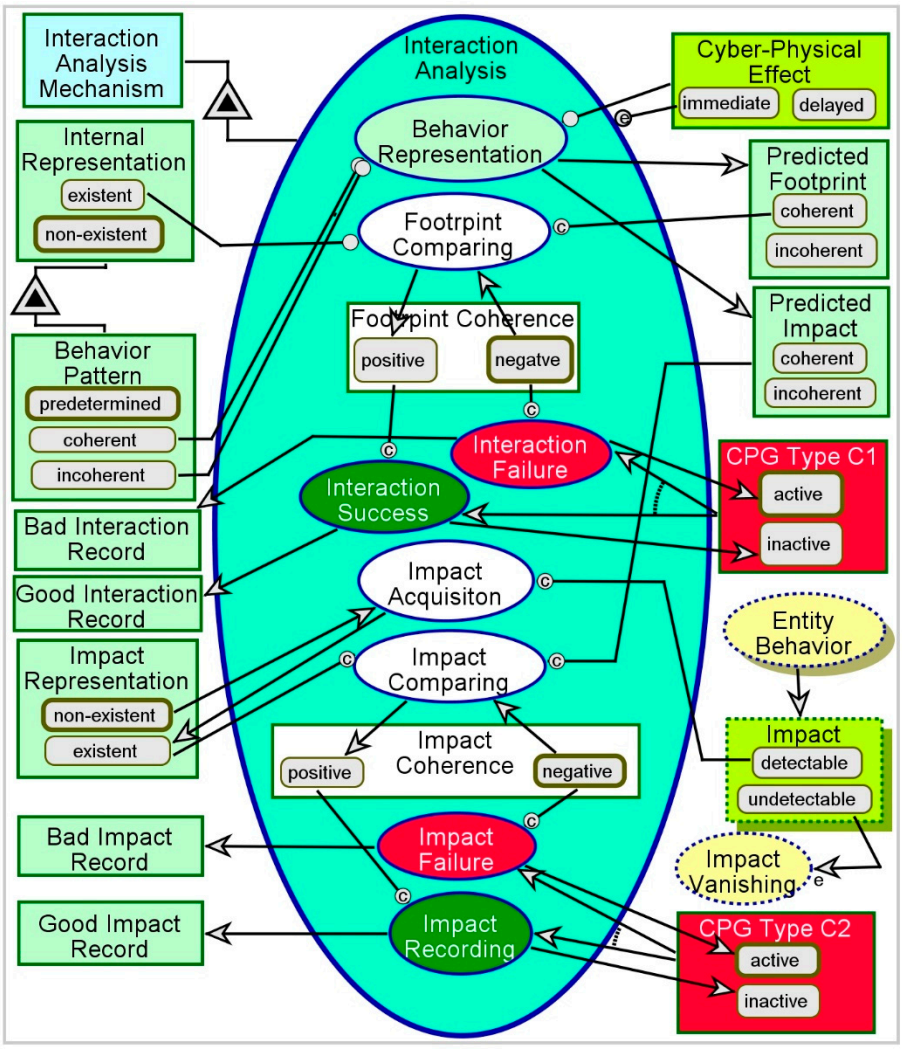


Figure 7. The Interaction Analysis process

Coherent representation may still result in incoherent interaction. This may imply that the interaction model is invalid – i.e., the system expects a result which is not feasible. For instance, imagine a central computer network control system that shuts down endpoint terminals to save energy. The control system orders some endpoint terminal to shut itself down when it is active – and known to be active by the central control – but the terminal does not shut itself down since a user working on that terminal manually disables that action on the terminal's side. In such a case, even though the representation is coherent, the interaction is not as expected, because the control system did not account for the possibility that a user may interfere with the interaction. Correct representation of the user as a separate and independent entity may help resolve this problem.

Interaction coherence and impact coherence are not completely dependent. Incoherent interaction may still result in a coherent impact if the interaction is within the tolerance boundaries for the impact to occur, or if the entity is intelligent or robust enough to compensate for the incoherent interaction. In addition, the desired result may be obtained by coincidence. For this reason, we do not set coherence values after interaction success or failure determination, as well as after impact success or failure determination. Rather, we issue a message or indication – **Good Interaction Record**, **Bad Interaction Record**, **Good Impact Record**, and **Bad Impact Record** – which can be used to drive a different process to investigate or find a root cause for any anomaly.

4. Applying CPGAME to the TMI2 Nuclear Reactor Meltdown Accident

In this section, we analyze the Three Mile Island nuclear reactor partial meltdown accident of March 28, 1979 (TMI2) according to the CPGAME approach. The Three-Mile Island, located three miles down the Susquehanna River from Middletown, Pennsylvania, hosts two nuclear pressurized water reactor units, which are active to this day. TMI2 is the severest

accident in the history of US commercial nuclear power plants. The main issue in this accident was CPG: a mismatch between the actual state of the physical system and its state as perceived by the control segment and consequently by the operators, based on their monitoring and control instruments. This mismatch caused the operators to take the wrong action based on their perception, which was not reflective of the reactor's actual state, and severely exacerbated the already dire situation, resulting in partial meltdown.

The course of events beginning on March 28, 1979, 04:00, is described in Table 3 [12], based on the description of events in US Nuclear Regulatory Commission website. We built an OPM model to reconstruct the reactor system as well as the failure scenario. This case study intends to demonstrate the applicability of OPM in facilitating CPG-aware system modeling, which significantly improves the ability to capture, explain, and predict catastrophic events and systemic CPG-related risk.

Table 3. Three-Mile Island 2 accident course of events [12]

#	Event	Effect
1.	On March 28, 1979, ~04:00, failure in secondary, non-nuclear section of plant, prevent main feedwater pumps from providing water to steam generators.	The steam generators cannot help cool the reactor core.
2.	The turbine-generator and reactor automatically shut down.	Pressure in primary, nuclear unit, begins to increase.
3.	Pilot-operated relief valve (PORV) opened.	Pressure drops.
4.	PORV closed.	PORV becomes stuck open.
5.	Instruments in control room indicate that PORV is closed.	Operators are unaware that cooling water is pouring out of stuck-open valve.
6.	Instruments in control room do not indicate how much water is covering the core.	Operators assume that as long as pressurizer water level is high, the core was properly covered with water.
7.	Alarm rings due to coolant loss, core exposure and overheating.	Operators do not identify loss-of-coolant accident.
8.	Water escaping through faulty PORV reduces pressure too much	Core is at risk of dangerous vibrations.
9.	Operators reduce emergency coolant input to primary unit.	Core is starved of coolant and overheats.
10.	Without sufficient cooling water, the nuclear fuel overheats	Nuclear fuel pellet cladding ruptures and they start melting.
11.	Someone notices another indicator of stuck-open PORV, closes emergency valve	Cooling water stops pouring out of reactor; reactor gradually stabilized.

4.1. Modeling the TMI Reactor System

We construct the model of the TMI reactor system and TMI2 accident by gradually evolving it through three model versions:

V1: The first, naïve model version, is a nominal model of the reactor system as it functions appropriately and without major issues.

V2: The second, fault-aware model version, extends the nominal model to cover the failure modes that led to the accident, but cannot cover or predict the crisis scenario due to indifference to the CPG.

V3: The third, CPG-aware model version captures the possible mismatch between the actual and perceived states of the system, and the wrong decisions that were made during the handling of the crisis and resulted in intensifying it.

We compare the three versions and show how CPGAME upgrades the ability to capture and simulate the CPG that caused TMI2. All the versions of the model are available on-line5 for readers who are interested in further experimenting with the models and analyzing the results.

[5] The TMI2 OPM Model Repository is available at https://1drv.ms/f/s!AsN2SH2tvCOWjmAwBGtBcUqlzcAF. Usage is permitted under the terms of the CreativeCommons CC BY-NC 4 license.

4.2. *The Naïve Model*

The graphical view of the TMI Reactor naïve model is shown in Figure 8. The textual OPL specification of the model, which is equivalent to the graphical view, is shown in Table 4. The current version of the model covers the nominal, failure-free operation of a **Pressurized Water Reactor**, and its main function, **Electric Energy Generating**. Each iteration of this cyclic process consists of the following four stages: 1) **Controlled Nuclear Reaction**, which transforms **Nuclear Fuel** to **Heat Energy**; 2) **Steam Generating**, which transforms the **Heat Energy** to **Steam**; 3) **Turbine Spinning**, which transforms **Steam** to **Mechanical Energy**; and finally, 4) **Electricity Generating**, which transforms **Mechanical Energy** to **Electric Energy**. The model is captured in simulation mode, while the fourth and final stage in the cycle is executing. The naïve model covers the ***Nominal Action***[6], which is the default option of the ***System***'s ***Action***, as shown in the pattern model in Figure 6. Currently the model does not account for any sensory activity.

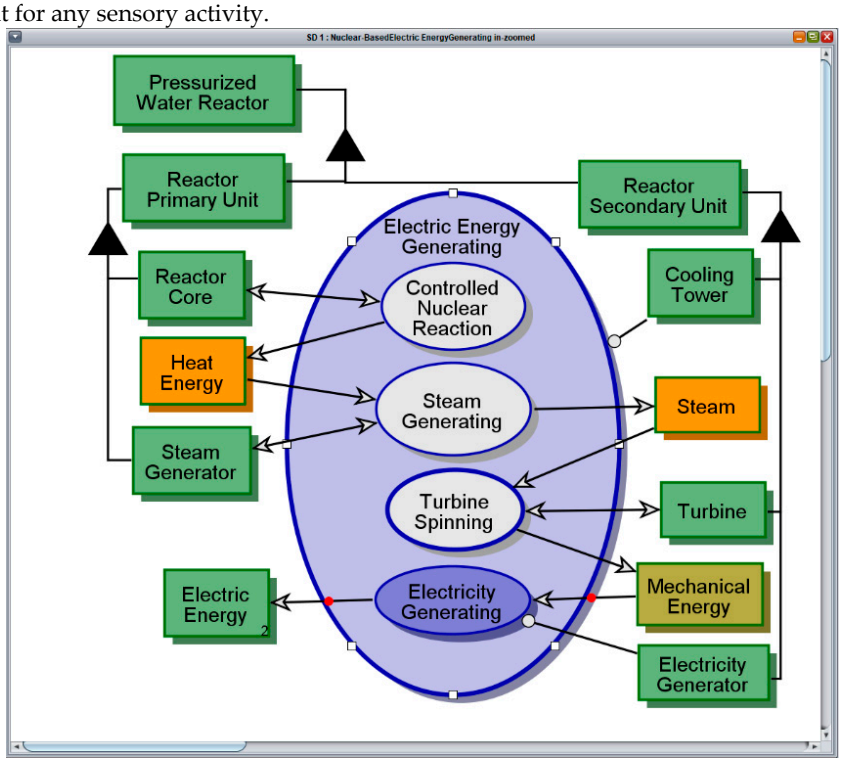


Figure 8. Electric Energy Generating using a pressurized water reactor – nominal operation

[6] References to things in the generic pattern model are italicized to distinguish them as pattern-related.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2017 doi:10.20944/preprints201706.0113.v1

Peer-reviewed version available at Sensors 2017, 17, 1644; doi:10.3390/s17071644

Table 4. Three-Mile Island Nuclear Reactor – Naïve Model – OPL Specification

ID	Statement
1000	Electric Energy is physical.
2000	Electric Energy triggers Electric Energy Generating.
3000	Steam is physical.
4000	Heat Energy is physical.
5000	Mechanical Energy is physical.
6000	Feedwater is physical.
7000	Feedwater can be cooling tower, condensor, or steam generator.
7100	cooling tower is initial.
8000	Pressurized Water Reactor is physical.
9000	Pressurized Water Reactor consists of Reactor Secondary Unit and Reactor Primary Unit.
9100	Reactor Secondary Unit is physical.
9200	Reactor Secondary Unit consists of Cooling Tower, Turbine, Electricity Generator, and Main Feedwater Pump.
9210	Cooling Tower is physical.
9220	Cooling Tower consists of Circulating Water Pump.
9221	Circulating Water Pump is physical.
9230	Turbine is physical.
9240	Turbine consists of Condensate Pump.
9241	Condensate Pump is physical.
9250	Electricity Generator is physical.
9260	Main Feedwater Pump is physical.
9300	Reactor Primary Unit is physical.
9400	Reactor Primary Unit consists of Reactor Core and Steam Generator.
9410	Reactor Core is physical.
9420	Steam Generator is physical.
10000	Nuclear Fuel is environmental and physical.
11000	Electric Energy Generating is physical.
12000	Electric Energy Generating consists of Controlled Nuclear Reaction, Steam Generating, Turbine Spinning, and Electricity Generating.
13000	Electric Energy Generating requires Cooling Tower, Pressurized Water Reactor, and Electric Energy.
14000	Electric Energy Generating yields Electric Energy.
15000	Electric Energy Generating zooms into Controlled Nuclear Reaction, Steam Generating, Turbine Spinning, and Electricity Generating.

ID	Statement
15100	Controlled Nuclear Reaction is physical.
15200	Controlled Nuclear Reaction requires Nuclear Fuel.
15300	Controlled Nuclear Reaction affects Reactor Core.
15400	Controlled Nuclear Reaction yields Heat Energy.
15500	Steam Generating is physical.
15600	Steam Generating affects Steam Generator.
15700	Steam Generating consumes Heat Energy.
15800	Steam Generating yields Steam.
15900	Turbine Spinning is physical.
16000	Turbine Spinning consists of Turbine Water Circulating, Water Cooling, Turbine Heat Removing, and Steam Generator Water Circulating.
16100	Turbine Spinning affects Turbine.
16200	Turbine Spinning consumes Steam.
16300	Turbine Spinning yields Mechanical Energy.
16400	Turbine Spinning zooms into Water Cooling, Turbine Water Circulating, Turbine Heat Removing, and Steam Generator Water Circulating.
16410	Water Cooling is physical.
16420	Water Cooling consumes Steam.
16430	Water Cooling yields cooling tower Feedwater.
16440	Turbine Water Circulating is physical.
16450	Turbine Water Circulating requires Circulating Water Pump.
16460	Turbine Water Circulating changes Feedwater from cooling tower to condensor.
16470	Turbine Heat Removing is physical.
16480	Turbine Heat Removing requires condensor Feedwater.
16490	Turbine Heat Removing yields Mechanical Energy.
16500	Steam Generator Water Circulating is physical.
16510	Steam Generator Water Circulating requires Main Feedwater Pump and Condensate Pump.
16520	Steam Generator Water Circulating changes Feedwater from condensor to steam generator.
16600	Electricity Generating is physical.
16700	Electricity Generating requires Electricity Generator.
16800	Electricity Generating consumes Mechanical Energy.
16900	Electricity Generating yields Electric Energy.

4.3. *The Fault-Aware Model*

Having constructed the nominal reactor model, we gradually extend it to cover the possible faults and failures that led to the meltdown accident. This stage can help visualize and simulate first-order failure modes, but it does not yet make any distinction between the physical failure and its identification. It is still assumed that a physical fault is directly and immediately identified by the system. To some extent, this is still a naïve approach, as it ignores the perception gap, but the model is still more informed than the nominal model.

We relax the assumption that resources, instruments, inputs, or outputs in the model are always in a nominal state. A fault-aware in-zoomed view of **Steam Generating** from Figure 8 is shown in Figure 9. The disrupted objects, states, and processes are painted in red. Most of the fault-aware **Steam Generating** now specifies possible failure modes and anomalies. This view intentionally captures states and activities that are intuitively not supposed to be in a model: one does not expect a meltdown event, for instance, in a nuclear reactor's functional model. The concern raised by the addition of such a disturbing possibility to the model is secondary to the insight

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2017 doi:10.20944/preprints201706.0113.v1

Peer-reviewed version available at Sensors 2017, 17, 1644; doi:10.3390/s17071644

generated by understanding the impacts of such adverse events while analyzing the model. Another example is the **stuck-open** state of the **Pilot-Operated Relief Valve (PORV)**. It may imply component reliability issues, which designers often prefer to conceal. Highlighting such an issue is exactly how the model enhances overall system reliability and provides important information on critical failure modes. Note that the *possibility* of a failure or mismatch is what interests and intrigues us, more than its *probability*.

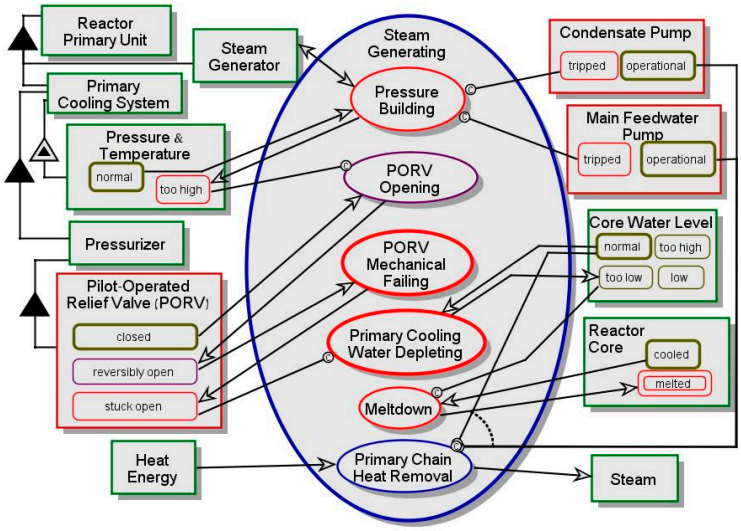


Figure 9. Steam Generating – Fault-Aware Model

4.4. *The CPG-Aware Model*

The CPG was a critical factor in the TMI nuclear reactor operators' decision to decrease water supply to the reactor core. The operators were not aware of the fact that after the PORV stuck, water was still pouring out of the reactor core, causing coolant starvation and reactor overheating. Thinking instead that there is excess water due to the wrong PORV indication, in order to avoid dangerous core vibrations, they shut down whatever emergency water that was still flowing to cool down the core, sealing its fate.

Figure 11 shows a screenshot of running OPCAT simulation of the CPG-aware model, while the **PORV Operating** process is executing. This process is a more robust replacement for **PORV Mechanical Failing** in the fault-aware model in Figure 9, taking place after **PORV Opening**. During the process, the **PORV** becomes **stuck-open**, instead of **closed**, due to its **fault-prone PORV Condition**. At the same time, the **PORV Indicator** reads **open**, but the **Determined PORV Status** is set to **closed** by mistake. The **Determined PORV Status** constitutes an ***Internal Representation*** of the corresponding attribute of the **PORV**, which is an ***External Entity*** for the control system. In this case there is a mismatch between the actual state of the physical entity and the perceived state of the representation. We have classified this situation as ***CPG Type B1***. If the **Determined PORV Status** were set to **open**, regardless of the state of **PORV Indicator**, it would have led to **Secondary PORV Closing**, and the control team could have saved the day. This could have happened if the water that was flowing out through the PORV were monitored, rather than the mechanical state indicator. Hence, this is also a case of ***CPG Type A1*** with respect to the escaping water. This examples also highlights the importance of a) capturing the human operator's understanding of the situation, and not only of the output that is provided by the system to the operator, and b) specifying viable, robust, and reliable detection solutions that would provide direct rather than indirect indications.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2017 doi:10.20944/preprints201706.0113.v1

Peer-reviewed version available at Sensors 2017, 17, 1644; doi:10.3390/s17071644

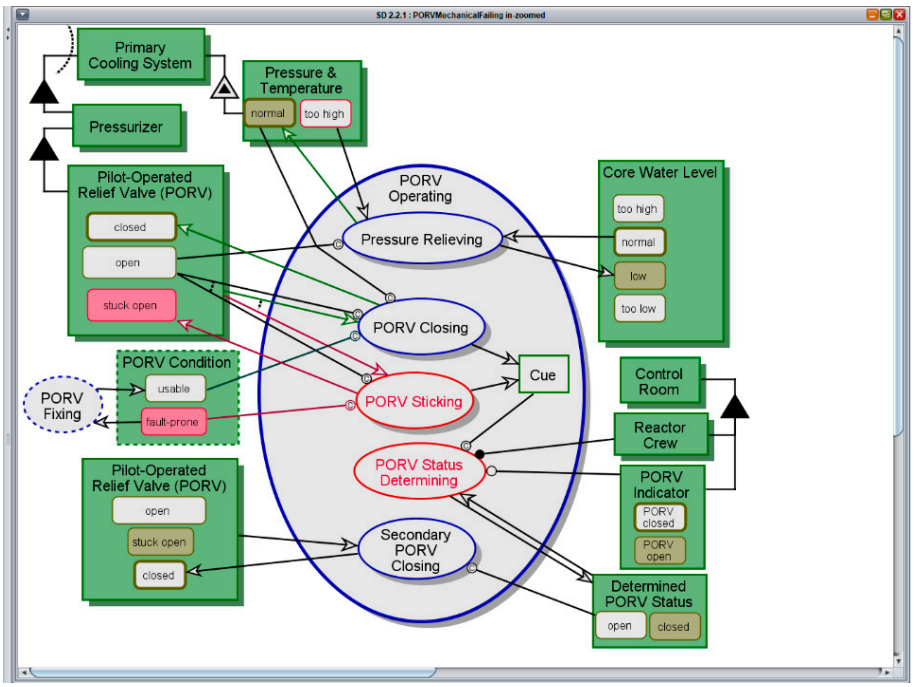


Figure 10. PORV Operating - CPG-Aware Model

The CPG in the TMI2 accident was in fact double. After the PORV indication was misread, a wrong conclusion was made about the amount of water in the reactor, relying on the assumption that since the PORV is closed then the amount of water is probably sufficient. No sensor for direct measuring of the water level in the core was used. The water level was wrongly assessed based on indirect indication.

In Figure 11 we zoom into **Primary Cooling Water Controlling**, a more robust replacement of **Primary Cooling Water Depleting**, which appears in Figure 9 after **PORV Mechanical Failing**, and follows **PORV Operating**. This process takes place only if the **PORV** is **stuck-open**, hence this is a ***Disrupted Action***. First, the **Determined PORV Status** reads **closed**, so **Core Water Level Determining** occurs and sets **Determined Core Water Level** to **too high**, while it is in fact **normal** or even **low**. This is the second ***CPG Type B1***, in which the wrong estimation of the water level was reached. If the **Determined PORV Status** had read **open**, the **Emergency Water Supplying** process could have taken place, the actual **Core Water Level** could have been balanced, and safety could have been restored. However, setting **Determined Core Water Level** to **too high** caused the opposite – **Emergency Water Supply Stopping** occurs, causing the **Core Water Level** to be **too low**, and the **Pressure & Temperature** to be **too high**. This is also both a case of ***CPG Type B2***, due to the failure to simulate and predict the behavior of the system once water supply is stopped, and also a case of ***CPG Type C1***, due to the failure to perform the intended interaction with the reactor and the water.

The result is shown in Figure 12, which captures the model's running simulation while executing the **Meltdown** process, which only takes place if **Core Water Level** is **too low**. This situation is never supposed occur, but we can simulate and mitigate it by improving the design thanks to the CPG-aware modeling and simulation. The **melted Reactor Core** state is a manifestation of ***CPG Type C2*** – failure to obtain the intended impact on the physical system.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2017 doi:10.20944/preprints201706.0113.v1

Peer-reviewed version available at Sensors 2017, 17, 1644; doi:10.3390/s17071644

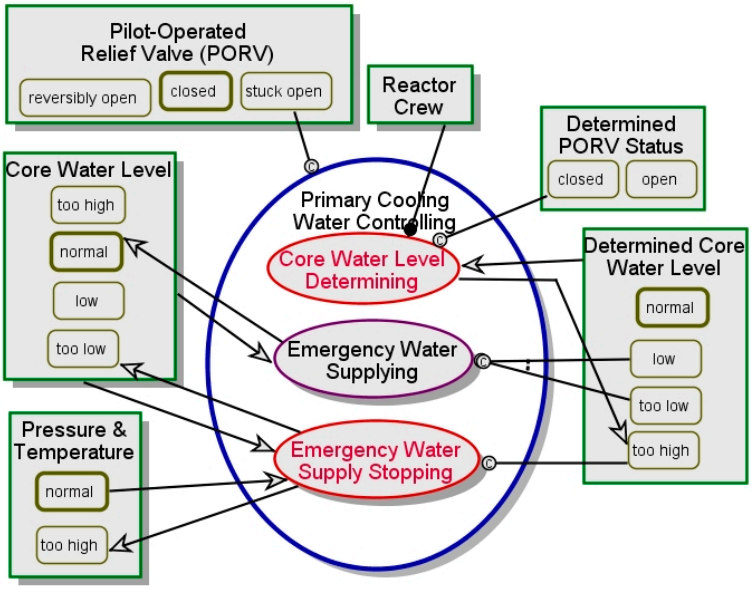


Figure 11. Primary Cooling Water Controlling – CPG-Aware Model

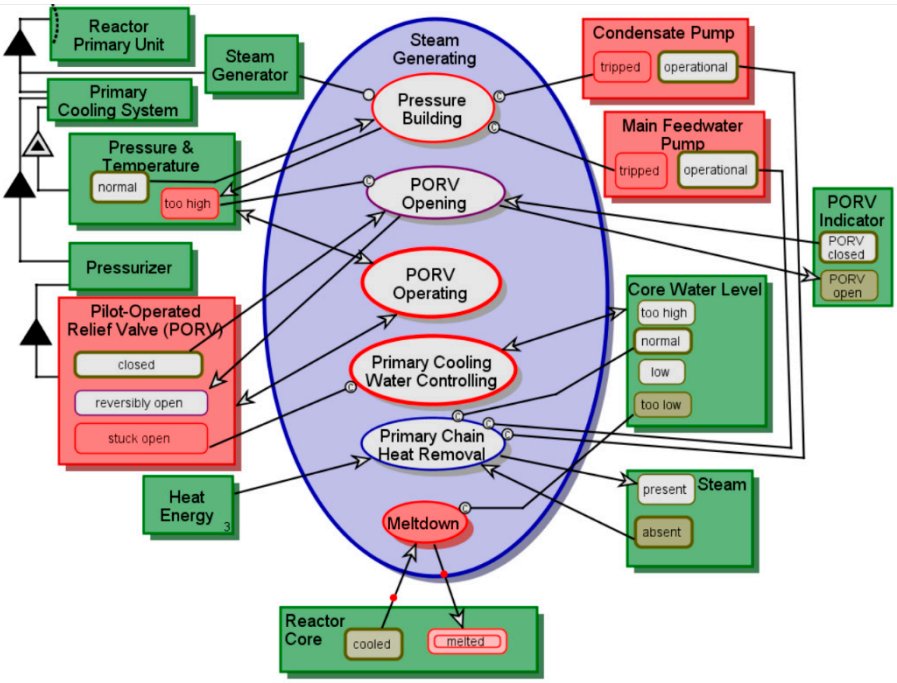


Figure 12. Steam Generating – Meltdown – CPG-Aware Model

4.5. *Enhanced Model Evaluation*

We evaluate the contribution of the CPG-aware OPM model relative to the nominal model through several perspectives. First, we compare the number of statements in each model in order to determine the rate of improvement in the informativity of the model. A comparison of the V1 and V3 is summarized in Table 5. V2 is an interim version, and is therefore omitted from the comparison. The total number of statements in V3 (186) more than tripled itself compared to V1 (60). Especially noticeable is the growth in the number of behavioral statements (from 25 to 97), due to the CPG-associated procedure and conditionality specification in the context of the stuck-open PORV and the adverse results. Two specific statement kinds with notable growths are State-set

Definition (from 1 to 13) and Condition Link (from 0 to 18). The growth in states and conditions marks the focus shift from general structure and process specification in V1 to situational and conditional modeling in V3, and highlights the evolvability of the same OPM model to cover these aspects. In addition, the model revision started as a focused elaboration on a specific failure mode (stuck-open PORV), but necessitated many additional and complementary modifications and extensions to cover the problem and course of events that we were trying to capture. This simple comparison clearly shows that a CPG-aware model is significantly more informative than its nominal counterparts, if only due to the idea that disruption-informed modeling is self-expanding, as multiple implications, considerations, and complementary aspects arise once the model is constructed this way.

Table 5. TMI2 Comparative Analysis of Model Versions

Measure	Nominal Version (V1)	CPG-Aware Version (V3)	Growth Rate
Total statements	60	186	+126 (210%)
Structural statements	35	89	+54 (154%)
• **State-set Definition**	1	13	+12 (1200%)
Behavioral statements	25	97	+72 (288%)
• **Condition Link**	0	18	+18

The TMI2 CPG-aware model directly covers 5 of 6 CPG types: A1, B1, B2, C1, and C2. CPG Type A2 is covered implicitly or indirectly by this example, since the reactor system's failure to detect the escaping water keeps the CPG Type A2 active by default. Table 6 summarizes the six CPG types and how they were demonstrated in the TMI2 example.

Table 6. TMI2 model coverage of CPG cases

CPG Type	Demonstrated	How / Why
A1 (No Detection)	Yes	Water escaping through PORV not detected.
A2 (No Acquisition)	Indirectly	Water escaping through PORV not acquired.
B1 (State Representation)	Yes	Determined PORV status vs actual PORV status Determined water level vs actual PORV status
B2 (Behavior Representation)	Yes	Predicted water and core behavior due to emergency water supply stopping
C1 (Interaction)	Yes	Water level depleting, rather than steadying, following emergency water supply stopping
C2 (Impcat)	Yes	Meltdown, rather than core stabilizing

5. Conclusion and Discussion

The CPG pattern establishes a well-defined distinction between an entity—a user or actor, another subsystem, an asset, or a resource—and its representation. One should bear in mind that the entity in the model is already a representation of either the real entity or the informatical representation, so the additional abstraction and distinction layer has not been trivial to enact until now. The CPGAME approach facilitates representation-based interaction between the system and the external entity. This pattern provides for modeling and subsequent model-based handling of anomalies, such as lost, untracked, or misperceived physical objects. The more the cyber-physical environment is unreliable and inconsistent, the more critical is applying the CPGAME approach. Although CPGAME add a layer of complexity to the system, accounting for CPG increases the accuracy, fidelity, reliability, safety, and security of the system, and consequently the overall performance level of systems in general and safety-critical systems in particular.

Integrating CPG-aware design elements into existing system models is challenging. As we show in the TMI2 case study, OPM facilitates extending nominal models to make them CPG-aware. Ignoring CPG-related problems or failure to address them during design time gives them a "green light" to show up during system operation, often in the worst time possible and with potentially devastating consequences. The TMI2 case is a prime example of CPG in complex CPSs, and of the significant impact that CPGAME has on predictability and mitigation of risks and other adverse effects. It also shows that CPG may appear also in legacy systems, and this notion should receive special attention from owners of similar legacy systems.

Future research involves the study of additional cases in which the CPG was or may have been a primary factor for systems' dysfunctional or disrupted behavior. One such case, the Malaysia Airlines 370 disappearance, is obviously related to CPG in the air traffic control systems that should have followed the flight when it disappeared. While a thorough audit of this case would require access to confidential or classified control systems and records, the theoretical explanation of this case in light of the CPG and using CPGAME would be a major contribution to the investigation of the case by air safety authorities and agencies, and would benefit stakeholders in similar conditions around the globe.

In addition, we intend to demonstrate CPGAME on a variety of system analysis cases, and in addition review the designs of existing systems in order to determine whether they are exposed to CPGs and whether that exposure could cause an impact on system behavior or its outcomes. In such a case, we would propose ways and mechanisms to mitigate the potential adverse impacts of the CPG.

Acknowledgements: We thank Gordon Center for Systems Engineering at Technion – Israel Institute of Technology for supporting this research.

References and Notes

1. Lee, E. a. Cyber Physical Systems: Design Challenges. *11th IEEE Int. Symp. Object Component-Oriented Real-Time Distrib. Comput.* **2008**.
2. Tan, Y.; Goddard, S.; Pérez, L. C. A prototype architecture for cyber-physical systems. *ACM SIGBED Rev.* **2008**, *5*, 1–2.
3. Kolin, K. Philosophy of Information and the Fundamentals of Informatics. In *The Third International Conference "Problems of Cybernetics and Informatics"*; Baku. Azerbaijan, 2010; pp. 290–293.
4. Hayles, K. *How we became posthumans*; The University of Chicago Press, 1999.
5. Mizzaro, S. Towards a theory of epistemic information. *Inf. Model. Knowl. Bases* **2001**.
6. Hayes, I.; Jackson, M.; Jones, C. Determining the specification of a control system from that of its environment. In *Lecture Notes in Computer Science: FME 2003: Formal Methods*; Araki, K.; Gnesi, S.; Mandrioli, D., Eds.; Springer, 2003; pp. 154–169.
7. Mordecai, Y.; Chapman, C.; Dori, D. Conceptual Modeling Semantics for the Physical-Informatical Essence Duality Problem. In *IEEE International Conference on Systems, Man, and Cybernetics - SMC2013*; IEEE: Manchester, UK, 2013.
8. Mordecai, Y.; Orhof, O.; Dori, D. Modeling Software Agent Awareness of Physical-Informatical Essence Duality. In *IEEE International Conference of Software Science, Technology, and Engineering - SwSTE 2014*; IEEE: Ramat Gan, Israel, 2014.
9. INCOSE *Systems Engineering Handbook*; Haskins, C.; Forsberg, K.; Krueger, M.; Walden, D.; Hamelin, R. D., Eds.; v. 3.2.2.; INCOSE, 2011.
10. Leveson, N. *Model-based analysis of socio-technical risk*; Working Paper Series; 2004.
11. Leveson, N. G. *Engineering a safer world*; MIT Press, 2011.
12. USNRC *Backgrounder on the Three Mile Island Accident*; 2014.
13. McNutt, M. *Science*. 2014, p. 947.
14. Rosenblueth, A.; Wiener, N. The role of models in science. *Philos. Sci.* **1945**, *12*, 316–321.
15. Haimes, Y. Y. *Risk Modeling, Assessment, and Management*; Third Edit.; John Wiley & Sons, 2009.
16. Dori, D. *Object-Process Methodology: A Holistic Systems Approach*; Springer-Verlag: Berlin, Heidelberg, 2002.
17. ISO/TC 184 ISO/PAS 19450L2015(en) Automation systems and integration — Object-Process Methodology 2014.
18. Embley, D.; Thalheim, B. *Handbook of conceptual modeling: theory, practice, and research challenges*; Embley, D.; Thalheim, B., Eds.; Springer, 2011.
19. Estefan, J. A. *Survey of Model-Based Systems Engineering Methodologies*; 2008.
20. Ramos, A. L.; Ferreira, J. V.; Barceló, J. Model-based systems engineering: An emerging approach for modern systems. *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.* **2012**, *42*, 101–111.
21. Hintikka, J. Individuals, possible worlds, and epistemic logic. *Nous* **1967**, *1*, 33–62.
22. *Handbook of knowledge representation*; Harmelen, F. Van; Lifschitz, V.; Porter, B., Eds.; Elsevier, 2008.
23. Wang, Y.; Kinsner, W.; Zhang, D. Contemporary cybernetics and its facets of cognitive informatics and

computational intelligence. *IEEE Trans. Syst. man, Cybern. Part B, Cybern.* 2009, *39*, 823–33.

24. Tomlin, C.; Member, S.; Pappas, G. J.; Sastry, S. Conflict Resolution for Air Traffic Management : A Study in Multiagent Hybrid Systems. **2000**, *43*, 509–521.
25. Leveson, N. *Engineering a safer world*; MIT Press, 2011.
26. Jaiganesh, V.; Mangayarkarasi, S.; Sumathi, P. Intrusion Detection Systems : A Survey and Analysis of Classification Techniques. **2013**, *2*, 1629–1635.
27. Mitchell, R.; Chen, I. A Survey of Intrusion Detection Techniques for Cyber-Physical Systems. *ACM Comput. Surv.* **2014**, *46*, 55:1-29.
28. Sinai, M. Ben; Partush, N.; Yadid, S.; Yahav, E. *Exploiting Social Navigation*; 2014.
29. Chmielewski, M.; Gałka, A.; Jarema, P.; Krasowski, K.; Kosiński, A. Semantic Knowledge Representation in Terrorist Threat Analysis for Crisis Management Systems. **2009**, 460–471.
30. Victor, G. J.; Rao, M. S.; Venkaiah, V. C. Intrusion Detection Systems - Analysis and Containment of False Positives Alerts. *Int. J. Comput. Appl.* **2010**, *5*, 27–33.
31. Carlson, D.; Altakrouri, B.; Schrader, A. AmbientWeb: Bridging the Web's cyber-physical gap. In *Internet of Things (IOT), 2012 3rd International Conference on the*; IEEE: Wuxi, 2012.
32. Maciejowski, J. M. Model discrimination using an algorithmic information criterion. *Automatica* **1979**, *15*, 579–593.
33. Goldstein, M.; Tech, V.; Va, B.; Rougier, J. Assessing Model Discrepancy Using a Multi-Model Ensemble. *Sci. York* **2008**, 1–35.
34 . Mordecai, Y.; Dori, D. Conceptual Modeling of System-Based Decision-Making. In *INCOSE Internaional Symposium*; INCOSE: Las-Vegas NV, USA, USA, 2014.
35. Mordecai, Y.; Orhof, O.; Dori, D. Model-Based Interoperability Engineering in Systems-of-Systems and Civil Aviation. *IEEE Trans. Syst. Man Cybern. Syst.* **2016**, *TBD*, TBD.
36. Mordecai, Y.; Raju, P.; Chapman, C.; Dori, D. Physical-Informatical Essence-Duality-Aware Generic Modeling of Threat Handling Processes. In *European Modeling Symposium - EMS2013*; IEEE: Manchester, UK, 2013; pp. 97–102.
37. Morris, B. A.; Harvey, D.; Robinson, K. P.; Cook, S. C. Issues in Conceptual Design and MBSE Successes: Insights from the Model-Based Conceptual Design Surveys. *INCOSE Int. Symp.* **2016**, *26*, 269–282.
38. Dori, D.; Linchevski, C.; Manor, R. OPCAT – An Object-Process CASE Tool for OPM-Based Conceptual Modelling. In *1st International Conference on Modelling and Management of Engineering Processes*; Heisig, P., Clarkson, J., and Vajna, S., Ed.; University of Cambridge: Cambridge, UK, 2010; pp. 1–30.
39. Dubois, D.; Prade, H. Representation and combination of uncertainty with belief functions and possibility measures. *Comput. Intell.* **1988**, *4*, 244–264.
40. Clemen, R.; Winkler, R. Combining probability distributions from experts in risk analysis. *Risk Anal.* **1999**, *19*.
41. Reich, Y.; Barai, S. V Evaluating machine learning models for engineering problems. **1999**, *13*, 257–272.