Conceptual Modeling Semantics for the Physical-Informatical Essence Duality Problem

Yaniv Mordecai
Faculty of Industrial & Management Engineering, Technion – Israel Institute of Technology, Haifa, Israel
yanivmor@technion.ac.il

Craig Chapman
Knowledge Based Engineering Lab, Birmingham City University, Birmingham, UK
craig.chapman@bcu.ac.uk

Dov Dori
Faculty of Industrial & Management Engineering, Technion – Israel Institute of Technology, Haifa, Israel;
Engineering Systems Division, Massachusetts Institute of Technology, Cambridge MA, USA
dori@mit.edu

***Abstract*—Physical-Informatical Essence Duality (PIED) is the parallel existence of an entity as both an independent, mostly physical object, and its informatical representation. System models tend to neglect or ignore this dual representation and the two parallel manifestations of model entities. This may result in erroneous modeling and realization, as representations can be partial and mistaken. Agent perception or capturing of an entity of interest is fundamental to the way it affects or interacts with the entity. The problem intensifies in models of complex systems and systems of systems, which must account for original embodiments and their multiple representations by various agents. This paper proposes a formal model-based approach and theory for detecting, capturing, representing, and controlling this physical-informatical duality in system models. We illustrate the proposed semantics via formal object-oriented epistemic logic semantics and via Object-Process Methodology (OPM), the emerging ISO 19450 standard for Model-based Systems Engineering.**

Keywords- Informatics; Model-Based Systems Engineering; Physical-Informatical Essence Duality; Agent-Oriented Architecture; Object-Process Methodology; Epistemic Logic; Knowledge Representation

I. INTRODUCTION

Systems and agents interacting with the external environment maintain a model of the environment, which facilitates their interaction with it. Several agents interacting with the same environment can each hold a different model of this environment. The role of informatical manifestations of "real-world" entities in natural, societal, and technical processes is perhaps one of the fundamental challenges Cybernetics and Informatics are currently facing [1]. Agents' representation of the external world can have varying levels of fidelity, which is related to the level of detail an agent is capable of processing.

As systems become ever more complex, multidisciplinary, and interconnected, their interactions with physical, biological, and social environments are constantly evolving, such that their execution and control autonomy increases. System architects increasingly follow robust and resilient design paradigms, which generate systems that adapt or are easily adjustable to a variety of environments and working conditions. These systems apply case-based functionality; they constantly monitor themselves, and act modularly in various component and interface configurations. Closed-circuit control systems have been present for almost a century, but many current systems are allowed to make autonomous decisions, while human operators are gradually being removed from the operational loop, or left in the loop mostly for monitoring and exception handling.

In the system-environment construct, environment entities have a dual nature. The entity is defined once by the original (most often physical) embodiment, and once by the representation thereof, as held by each relevant agent. The dual existence of the original-physical and representational-informatical is defined in this paper as Physical-Informatical Essence Duality (PIED). PIED exists in virtually any type of system, including natural and societal contexts, complex systems, and systems-of-systems.

This duality is well noted in the literature on Epistemic Logic, Knowledge Representation, and Artificial Intelligence, [2], [3], [4], [5], [6]. However, it is not well-embedded in systems engineering, and this delicate notion eludes many system engineers, which leads to erroneous modeling, realization, and system behavior. In Data Integration, multiple representations must be accounted for [7], but the interaction with the environment is not addressed. Common modeling languages like UML and SysML [8] lack the means to capture the duality. System Safety studies have shown that systemic misconceptions led to dire consequences and catastrophes [9].

The purpose of this paper is to highlight the notion of the PIED in systems and models of these systems. We define and analyze this duality and its implications on system models, and present a framework for capturing the compound essence of entities and for handling dual entities in new and existing system models. We propose a formalism to define essence duality, which consists of Epistemic Logic semantics, and is also object-oriented. In addition, we develop a conceptual modeling approach based on Object-Process Methodology (OPM) [10], a holistic framework for complex systems modeling, and emerging ISO 19450 standard.

The rest of this paper is organized as follows: Section II discusses and extends the concept of PIED and the challenges it

978-1-4799-0652-9/13/$31.00 ©2013 IEEE

poses. Section III provides a brief description of OPM. Section IV provides a model-based framework for compound essence modeling and handling, and Section V summarizes this paper and discusses future extensions.

II. Physical-Informatical Essence Duality (PIED)

In this section, we provide some basic formal definitions and understanding of PIED. We also define the formalism for describing this aspect, drawing on the semantics of Epistemic Logic, with notation adjusted for conceptual modeling of engineering systems.

Systems interacting with their environment invariably contain a model of the environment, and it is inherently structured with reference to this environment. In a system comprising several sub-systems and agents, each sub-system has its own model of the environment, and the models may not necessarily feature the same level of detail, fidelity, and precision. Each entity in the environment has to be represented by the agent affecting it or interacting with it. Each internal model of the entity, held by agents and sub-systems, constitutes an additional manifestation of the original entity that best serves the owning agent's objective. Thus, an entity may be manifested by multiple representations, as perceived, captured, or held by various agents in the domain.

There are two characteristics of the entity that are relevant in this context: *affiliation* and *essence*. *Affiliation* is a characteristic which denotes the entity as either a) original and environmental, i.e., pertaining to the environment, or b) representational and systemic, i.e., pertaining to the system. This duality of the *physical world* vs. the *cyber world*, two constituents of the *natural world*, is a fundamental principle in Cybernetics [2]. Intelligence provides for translation from one view to another. We will address this issue later in this paper.

The second characteristic, *essence*, denotes the entity as either physical or informatical. While original entities are mostly physical, and representational ones are consequently informatical, all combinations of *affiliation* and *essence* are applicable and valid for this theory. Consider, for instance, a physical model of a physical structure such as a bridge or an airplane. While the model is representational, it is still physical.

The dual existence of the original-physical and representational-informatical, which is defined in this paper as PIED, is well-noted in the literature. Hayles [3] provides an in-depth discussion on the implications of information-matter duality and dualism, including the abstraction of information from matter, the separation of information from the material source it reflects, and the separation of content from medium. This work follows Bar-Hillel's Semantic Information Theory, which distinguished the meaning of content payload from the meaning of the physical carriers, namely, of information from data [11]. The way an entity is captured, understood, or perceived by each agent in the system is defined by Mizzaro as "Epistemic Information" [4], i.e., information derived from knowledge and understanding. Such information may differ from agent to agent, and even among similar or identical agents.

PIED poses several challenges to system architects, analysts, and developers. System models tend to describe the environment as it is perceived by the system. This seems like a reasonable paradigm. However, when the system affects the environment to the extent that the environment changes the system or itself, the model has to capture this effect as well. However, this delicate aspect eludes many system engineers. PIED must be acknowledged and embedded in the cognitive processes guiding the modeling, design, and development of complex systems and interactions with the environment.

Once PIED is recognized and understood, the problem becomes one of defining the knowledge-base of the system, and each of its relevant constituent components, mechanisms, and agents, regarding the environment. Knowledge about the environment is required so that system agents can make decisions and perform actions. The knowledge-base affects the outcome of reaction, action, and interaction. It includes what the system and sub-systems need to know about the environment, and what they think, know, and think they know about the environment at any given moment. While these notions have been well defined and elaborated in the Epistemic Logic and Artificial Intelligence literature [6], it has not been sufficiently embedded into Systems Engineering.

Distributed and interoperable information systems, in which several federated systems or system components hold representations or records of the same entity (e.g., a patient and the patient's records held by various medical information systems and devices), must account for the multiple informatical representations of entities they should coordinate, synchronize, or integrate [7]. Cyber-physical constructs also consist of the actual physical entity, to which information holding components refer, and it must in general also be acknowledged and modeled. This complicates the problem of information integration due to the challenge of aligning information with 'real world' entities.

Traditional system modeling approaches often fail to provide the semantics needed to capture and formally represent PIED. Modeling languages grounded in the information systems domain mostly refer to entities as either having a physical essence or an informatical essence, but not both. Typical UML models, for instance, capture Actors as top-level external entities, interacting with the system at the Use-Case level [8]. However, the Actor is not associated with an internal representative Class, Actor designation is not replicated for subsystems at lower levels, and sub-systems are rarely defined as actors of other sub-systems. The simple notion that sub-systems in a system constitute mutual actors, is critical to the understanding of cross-structural processes in distributed systems, and especially in systems of systems. Thus, neither entities nor sub-systems are captured in a PIED-aware manner.

Models of complex systems almost invariably contain a significant information system model segment alongside a real-world model segment. Thus, instead of allocating different representations for a physical entity and its model reference, modelers usually represent the external physical entity and its informatical reference as one and the same. This modeling approach is correct if and only if each representing entity in the modeled system is informed in real time about the current state and any change in the physical real-world entity. This condition can be restricted for aspects that are relevant only for practical purposes, but even so, it is rarely truly satisfied.

When the agent's conceptions mismatch the actual nature of external entities (including other sub-systems), errors, failures, and hazards can arise. A patient's medical condition is paramount to the treatment the patient should get. However, the treatment decision is not directly based on the patient's real condition, but on the condition as captured by the diagnosing person or device, which must be as precise as possible. Indeed, cases in which wrong diagnosis results in maltreatment with severe consequences are not too infrequent.

With this in mind, we now turn to propose formalism for defining, describing, and understanding essence duality and its implications on system behavior and environmental effects. Epistemic Logic provides semantics of defining, modeling, and processing epistemic information – knowledge and belief [5]. The proposed formalism consists of Epistemic Logic semantics, enhanced by object-oriented notation. This notational fusion provides for associating representations and effects with their owning agents.

Let E constitute an original entity. An entity's representation $R(E)$ consists of two aspects: a) recognition of existence X, and b) Perceived State S, as defined in (1).

$$R(E) \equiv \langle X, S \rangle \tag{1}$$

Recognition of existence is dichotomous, so X, can assume only 0 or 1. Since the entity E may exhibit various attributes, S can assume a wide, combinatorial range of values and sub-states, according to the number of attributes of the entity that are captured in $R(E)$.

Let $A.R(E)$ denote the representation of E held by agent A. $A.R(E)$ can be defined only if E is found in A's knowledge-base: $A.K_X(E)$, i.e., E is known to A when it exists. The recognition of existence is denoted by x. $A.R(E)$ can assume a state s only if s is defined in A's knowledge-base. $A.K_S(E)$ is the set of E's states that are known to A. It depends on A's recognition of existence, $A.K_X(E)$, but it is also distinguished from it, since agents may know the entity, but not necessarily its state. These limitations are expressed in (2) using the conditional "|" sign.

$$A.R(E) = \langle x | A.K_X(E), s | s \in A.K_S(E) \rangle \tag{2}$$

The entity may contain sub-components, or exhibit sub-processes and attributes, collectively referred to as *characteristics*. Each characteristic (i) of the entity E, $E.C_i$, constitutes a bona fide entity, and can be referred to recursively via (1). The agent has to be aware of its existence, as indicated by $K_X(E.C_i)$, and of its possible states (referred to at *values,* and denoted by v in the case of a property), as listed in $K_S(E.C_i)$. This is expressed in (3).

$$A.R(E.C_i) = \langle x | A.K_X(E.C_i), \quad v | v \in A.K_S(E.C_i) \rangle \tag{3}$$

Representations of the same entity may often be integrated, synchronized, or compared, in order to improve their fidelity. A joint representation of two or more representations n = 1,.., N is defined in (4) as $R \mid \{R_1(E), .., R_N(E)\}$.

$$R \mid \{R_1, .., R_N\} \equiv \langle h_X(\forall A_n.R(E).X), h_S(\forall A_n.R(E).S) \rangle \tag{4}$$

Each representation aspect (existence and state) is fused using a dedicated fusion function (h_X and h_S respectively). These functions are defined by the agent who generates the integrated representation. The result of such knowledge integration may 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 distribution of possible states with probabilities that sum to 1. A survey of various information fusion methods from various sources with varying reliabilities is provided in [12]. Bayesian Inference is referred to as a best practice for this process. In a similar manner, agents can enrich their own representations with external representations. Conflicts may appear in the way the various agents perceive the same entity, and must be resolved in the fusion functions.

The agent's run-time learning and knowledge-base extension capabilities are critical for ensuring high representation fidelity. Otherwise, if the original entity evolves and extends its state space, the agent can no longer represent it correctly. Representations in software-based environments are not strictly informatical. An external electro-mechanical controller, for instance, constitutes a physical entity, and can be captured by a software simulator with a representation of both the static and dynamic aspects of the physical original controller. The representation has the capability to make decisions and carry out actions, imitating the real controller's work, with the exception of physical force application. We refer to the simulator's essence in this case as *virtual*. Virtual essence enables the distinction of static or predefined data or information (including manipulated or processed information) from an entity or representation defined in an informatical manner, but behaving like, or imitating a physical one.

Discerning representations in physical environments may require exercising some imagination. A bridge, for instance, is designed to withstand various load distributions. The actual load on the bridge is captured in the stresses and forces applied on the parts and materials the bridge is made of. These forces are encoded in a way which enables bridge parts to respond to physical force, exactly like it can be detected and captured by various load, vibration, and stress detectors and sensors mounted on the bridge. Therefore, the forces and stresses stemming from the interfacing of bridge and load objects exhibit essence duality: the real-world, physical embodiment, the representation as detected by the sensors, and the representation as captured (though somewhat elusively) by bridge parts and materials reacting to the applied forces. This example demonstrates the role and independent existence of information in nature [2].

When an agent is likely to affect or interact with the original entity E, the effect of the agent's action depends on the state of the entity as assumed by the agent. In turn, the entity responds to the effect according to its actual state. If the original entity's state is the same as the state perceived by the agent, based on its own representation, then a coherent response occurs. Otherwise, the response can be and often is incoherent, depending on the system's robustness or resilience to such anomalies. The result of the response is defined as a new original entity, and the representation process can reapply to it in its entirety. This causality is expressed in (5), where ξ stands for the coherence of the result of the entity's response function, $E.g$, to the agent's action function, $A.f$, which is applied according to parameters of the representation. The result depends on the match between the real state $E.S$ and the state perceived by the agent $A.R(E).S$.

$$\xi[E.g(A.f(R(E))] = \begin{cases} 1, & A.R(E).S = E.S \\ 0, & A.R(E).S \neq E.S \end{cases} \quad (5)$$

III. Object-Process Methodology (OPM)

OPM [10], is a holistic, integrated framework for complex dynamic systems modeling, design, development and verification. OPM captures the functional, structural, and procedural aspects of the system, in a single unified, dual graphical-textual view.

OPM building blocks are objects and processes, collectively called things. Objects are things that exist and can have states. Processes are things that occur and transform objects: they generate and consume objects, or change their state. OPM things' inherent *Affiliation* and *Essence* attributes mark their exhibitors as systemic or environmental, and as physical or informatical, respectively. OPM things are connected by two types of links: structural and procedural. Structural links specify relations between objects, or between processes. Conversely, procedural links connect processes with objects or object states.

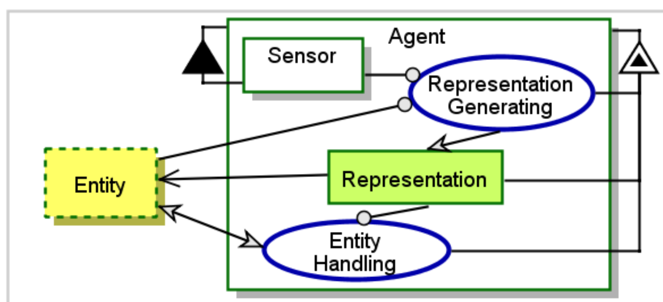
OPM models consist of a set of hierarchically organized Object-Process Diagrams (OPDs). The hierarchical structure alleviates system complexity through three mechanisms: (1) Structural hierarchies folding/unfolding; (2) Detailed behavior zooming-in/out; and (3) State expressing/suppressing. Each OPD is obtained by in-zooming or unfolding an object or a process in its ancestor OPD. Rectangles denote objects, ellipses denote processes, and round-angle rectangles ("rountangles") denote object-states, within the owning object's rectangle. The OPD hierarchical structure is accompanied by a corresponding set of structured textual model description sentences, written in Object-Process Language (OPL), a subset of English. OPCAT[1] [13] (OPM's free CASE tool) automatically generates OPL sentences in response to visual model edits. The textual description is equivalent to the graphical view, allowing for enhanced model understanding. Any model fact has to appear once in any diagram in the model, for it to be valid for the entire model.

OPM was selected as the basis for demonstrating PIED, due to several key features. The unified static-dynamic view has proven complexity management and alleviation capabilities that strongly support essence duality modeling [14]. Inherent complexity management via hierarchical decomposition enables concealing or revealing representations, as required for model understanding and communicating. The semantically equivalent graphical and textual views make OPM appealing to both sides of the human brain, catering to systems architects, domain experts, professionals and practitioners. The capability to generate meta-models—generic, multi-purpose models and patterns that can be instantiated and adapted for specific systems and problems—is a threshold condition for utilizing a language for metamodeling as practiced in this research. OPM's standardization as ISO 19450, as a basis for system and process modeling in ISO enterprise standards, enables accelerated dissemination of OPM for enterprise modeling. OPM's freely available CASE tool, OPCAT, fully supports the concepts and distinctions we explore, allows rapid demonstration and visualization of the problem, generates OPL texts automatically, and has the capability to capture and visualize complex interactions and dynamics with its built-in simulation engine. PIED-aware models recently submitted by undergraduate students validate OPM's applicability for PIED.

IV. Handling Essence Duality with OPM

In this section we describe an OPM metamodel for PIED-aware system models. The metamodel utilizes OPM's hierarchical in-zooming mechanism. Each level is captured in an OPD, followed by an equivalent, automatically generated OPL text, which was slightly edited for clarity. Natural language names of model artifacts replace the mathematical symbols used in section II, making the text easier to understand and read. Repeated occurrences of previously specified OPL statements are sometimes omitted.

The topmost level (Figure 1.) describes the interaction of an agent with an entity. The agent has to handle the entity, and for this purpose it owns an **Entity Handling** process. However, as our approach states, it must also hold a representation of the entity, which is generated by a **Representation Generating** process, which depends on the original entity.



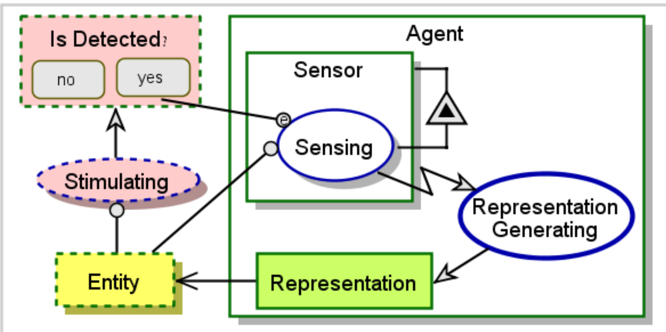
1. Entity is environmental and physical.
2. Agent is physical.
   2.1. Agent consists of Sensor.
      2.1.1. Sensor is physical.
   2.2. Agent exhibits Representation, as well as Entity Handling and Representation Generating.
   2.3. Representation Generating requires Sensor and Entity.
   2.4. Representation Generating yields Representation.
   2.5. Representation relates to Entity.
   2.6. Entity Handling requires Representation.
   2.7. Entity Handling affects Entity.

Figure 1. Top-Level Object-Process Diagram: Agent-Entity Interaction

The next level is obtained by in-zooming each process in the topmost level. Hence, it involves two diagrams: one for **Representation Generating** (Figure 2.) and one for **Entity Handling** (Figure 3.). **Representation Generating** depends on the ability of the agent's sensor to activate its **Sensing** function to detect the entity. However, **Sensing** cannot occur if it is not

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

stimulated. The occurrence of a **Stimulation** is an external random variable, which reflects the sensitivity of the sensor to the entity or change in an attribute value it is designed to measure, e.g., temperature. The entity's representation is generated in response to the detection of its existence, or a change in the value of a signifying attribute (which is not expressed in the metamodel but is done in a similar way).



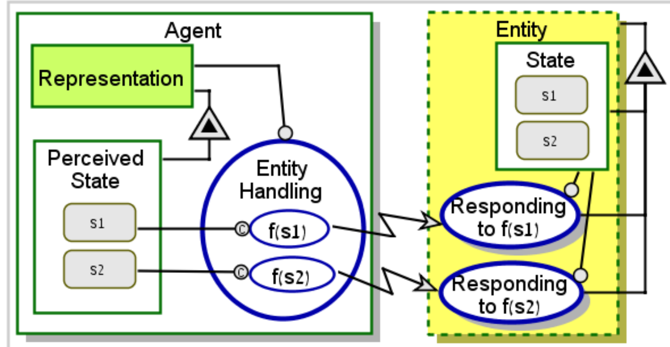
1. Stimulating is environmental and physical.
   1.1. Stimulating requires Entity.
   1.2. Stimulating yields Is Detected?.
2. Is Detected? is environmental.
   2.1. Is Detected? can be yes or no.
   2.2. Is Detected? triggers Sensing when it enters yes.
3. Sensor exhibits Sensing.
   3.1. Sensing is physical.
   3.2. Sensing requires Entity and yes Is Detected?.
   3.3. Sensing invokes Representation Generating.
   3.4. Representation Generating yields Representation.

Figure 2. Object-Process Diagram: Representation Generation

Entity Handling depends on the existence of the representation and of the state it assumes. **Entity Handling** is modified by the **Agent**'s **Representation**'s **Perceived State**, and there is a dedicated handling method for each perceived state **f(s$_i$)**. Each method triggers the respective **Response to f(s$_i$)** by the original entity, but this response also depends on the *original* **Entity**'s *actual* **State**, as elaborated in the next level.

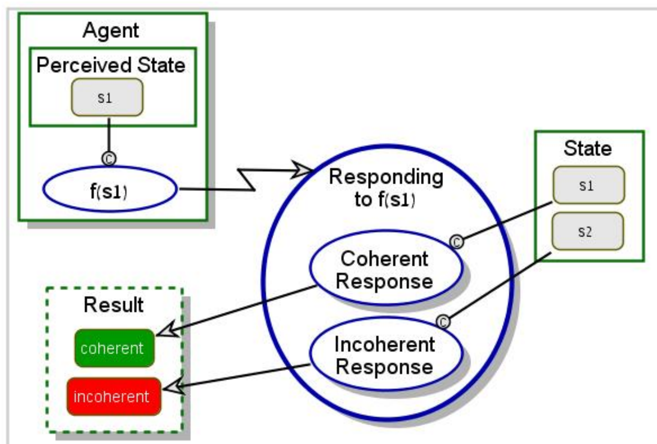
The entity's response to a specific action by the agent is modeled in Figure 4. We focus on the *coherence* of the result of the response as derived from the entity's actual state. If the entity is at the same state the agent thought it was when it triggered its action, then the **Result** is *coherent*, that is, the one that was expected by design. Otherwise, the response and the **Result** are *incoherent*. Some remedial action may be required to compensate for the potential harm done to the system and/or to its environment due to this incoherence.

The result of the interaction can affect the representation of the original entity. Detecting the result (with appropriate means), representing it, feeding it back into the system, and updating the original representation, are parts of a dedicated process, **Representation Refining**. Handling the result may include accepting or rejecting, and deciding whether the original entity's state needs adjustment. This may lead to restored or preserved coherence or escalated incoherence of the agent's (attempted) effect on the original entity. These concepts should be explored and formalized in future research.



1. Entity exhibits State, as well as Responding to f(s1) and Responding to f(s2).
   1.1. State can be s1 or s2.
   1.2. Responding to f(s1) requires State.
   1.3. Responding to f(s2) requires State.
2. Representation exhibits Perceived State.
   2.1. Perceived State can be s1 or s2.
3. Entity Handling consists of f(s1) and f(s2).
   3.1. Entity Handling requires Representation.
   3.2. f(s1) occurs if Perceived State is s1.
   3.3. f(s1) invokes Responding to f(s1).
   3.4. f(s2) occurs if Perceived State is s2.
   3.5. f(s2) invokes Responding to f(s2).

Figure 3. Object-Process Diagram: Entity Handling



1. Perceived State is s1.
2. Result is environmental and physical.
3. Result can be coherent or incoherent.
4. f(s1) occurs if Perceived State is s1.
5. f(s1) invokes Responding to f(s1).
6. Responding to f(s1) consists of Coherent Response and Incoherent Response.
   6.1. Coherent Response is physical.
      6.1.1. Coherent Response occurs if State is s1.
      6.1.2. Coherent Response yields coherent Result.
   6.2. Incoherent Response is physical.
      6.2.1. Incoherent Response occurs if State is s2.
      6.2.2. Incoherent Response yields incoherent Result.

Figure 4. Object-Process Diagram: Entity Responding to Agent Action

The design pattern introduced in this section is a simple, clear way for capturing an entity's dual essence. For the sake of clarity, we did not name the entity and the representation by the same name. However, in practical modeling, both the entity and its representation can be named identically, similarly, or inclusively. Including the entity's name and another word indicating representativeness in the name of the representation is a good practice. Suffixes like the derivative tag ('), the words "record", "view", "representation" (for object entities), "reflection", "simulation" (for process entities), or the prefix "my" often used by software designers and programmers, are common examples.

V. SUMMARY

This paper discusses the importance of understanding, identifying, and capturing Physical-Informatical Essence Duality (PIED) of entities in systems and models. We have described a formal, model-based framework for dealing with the challenges of recognizing and containing PIED. Lack of attention for this intricacy can lead to misconceptions and modeling errors which cannot be mitigated unless the PIED is well defined. System agents and components trying to affect an external entity using what they think they know about that entity, can lead to incoherent results due to mismatch between entities' actual and perceived states, resulting in component or system failures across a wide spectrum of severities.

Future work will extend the PIED-aware model with layers reflecting aspects such as feedback, conflict resolution, refinement and fusion of representations, and embedded coherence analysis. Complementary modeling of PIED for processes is also required. We intend to incorporate PIED and related notions presented in this paper in various system design problems, especially in the fields of defense, health, and sustainability. Existing OPM models can be easily and methodically extended to capture and contain PIED situations. At first, such extensions complicate the models since a lot of entities are multiplied and the entity-representation gap must be abridged, but this eventually makes the extended model and the system much more robust and resilient.

ACKNOWLEDGMENT

The authors would like to thank the anonymous referees for their useful feedback and advice on this paper. This research was carried out with the generous financial support of the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement No. 262044 – VISIONAIR, the Bernard M. Gordon Center for Systems Engineering at the Technion, the Jacobs Technion-Cornell Innovation Institution at NYC, and the Knowledge Based Engineering Lab at Birmingham City University, UK.

REFERENCES

[1] K. Kolin, "Philosophy of Information and the Fundamentals of Informatics," in *The Third International Conference "Problems of Cybernetics and Informatics,"* 2010, pp. 290–293.

[2] Y. Wang, W. Kinsner, and D. Zhang, "Contemporary cybernetics and its facets of cognitive informatics and computational intelligence.," *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics*, vol. 39, no. 4, pp. 823–33, Aug. 2009.

[3] K. Hayles, *How we became posthumans*. The University of Chicago Press, 1999.

[4] S. Mizzaro, "Towards a theory of epistemic information," *Information Modelling and Knowledge Bases*, 2001.

[5] J. Hintikka, "Individuals, possible worlds, and epistemic logic," *Nous*, vol. 1, no. 1, pp. 33–62, 1967.

[6] F. Van Harmelen, V. Lifschitz, and B. Porter, Eds., *Handbook of knowledge representation*. Elsevier, 2008.

[7] F. Saltor, M. Castellanos, and M. García-Solaco, "Suitability of datamodels as canonical models for federated databases," *ACM SIGMOD Record*, pp. 44–48, 1991.

[8] L. Favre, *UML and the Unified Process*. IRM Press, 2003.

[9] N. Leveson, *Engineering a safer world*. MIT Press, 2011.

[10] D. Dori, *Object-Process Methodology: A Holistic Systems Approach*. Springer, 2002.

[11] Y. Bar-Hillel and R. Carnap, "Semantic Information," *The British Journal for the Philosophy of Science*, vol. 4, no. 14, pp. pp. 147–157, 1953.

[12] R. Clemen and R. Winkler, "Combining probability distributions from experts in risk analysis," *Risk Analysis*, vol. 19, no. 2, 1999.

[13] D. Dori, C. Linchevski, R. Manor, and O. M. Opm, "OPCAT – An Object-Process CASE Tool for OPM-Based Conceptual Modelling," in *1st International Conference on Modelling and Management of Engineering Processes*, 2010, pp. 1–30.

[14] D. Dori, I. Reinhartz-berger, and A. Sturm, "Developing Complex Systems with Object-Process Methodology Using OPCAT 1 The Basis□: Object-Process Methodology," pp. 570–572, 2003.