Presence-Awareness: A Conceptual Model-Based Systems Biology Approach

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

Judith Somekh
Faculty of Industrial Engineering and Management
Technion – Israel Institute of Technology, Haifa, Israel
ysomekh@gmail.com

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

***Abstract*—We propose a model-based systems engineering approach for representing and analyzing the perception and consideration of the presence of external entities by systems. This approach is inspired by the conceptual model-based systems biology paradigm we have previously proposed. The notion of presence and absence of surrounding objects, or occurrence of surrounding processes, is becoming ever more critical in a world of highly-aware automation agents and the increasingly complex and stochastic environmental configurations in which these agents act. This problem is well-known in the biological modeling domain. System models usually focus on nominal, mainstream aspects and scenarios. Presence-related aspects are often assumed to pose no issue, and remain unattended in both the conceptual and actual system design phases. The Pareto rule-of-thumb of system design asserts that about 20% of design and development efforts cover the nominal, "sunny day" process, while the other 80% cover exceptions, alternative scenarios, failure modes, edge conditions, special settings, and emergencies. We propose an approach to extend the naïve, nominal system model by enriching it with notions of presence-awareness and sensitivity. This approach is facilitated by Object-Process Methodology, a holistic system and process modeling framework, currently being adopted as ISO standard 19450.**

Keywords—Model-based Systems Engineering; Exception Modeling; Object-Process Methodology; Conceptual Model-based Systems Biology; Input Variability; Process Anomaly; Exception Handling; Presence; Presence-Awareness

I. INTRODUCTION

Conceptual models of phenomena and systems have a central role in science, research and development, engineering, and operation. The more indecipherable the system, the more degrees of modeling and experimentation are required to understand it [1]. This is especially true in situations and contexts of high and extreme uncertainty [2]. However important models are for understanding of real world static and dynamic system aspects, they capture but a small portion of the problem domain. Models tend to focus on the primary, standard, nominal, typical, optimistic "sunny day scenario" aspects, where all goes well as planned. This focus serves to clarify the "big picture" and core of ideas behind the modeled system. Alternative scenarios, variant structures or settings, edge conditions, exceptions, failure modes, etc. are usually defined externally, outside of the core system model. At best, in most cases they are described in separate models, based on dedicated modeling techniques – especially in risk analysis, exception handling, and non-linear system control.

According to the Pareto Rule of Thumb, about one fifth of the effort is spent to create four-fifths of the total value, and vice versa. It would not seem unreasonable to claim that core functionality requires a significantly smaller proportion of design and development, compared to the remainder of enveloping, variant, exceptional, or extended functionality. Yet, neglecting these aspects can prevent the system from fulfilling its primary purposes due to failure to cover exceptional situations. Models can and should provide additional valuable information and insights by including all those non-trivial aspects, which are often elusive and misunderstood by many stakeholders who are not subject-matter experts. Such aspects can be well understood as part of a system model.

One of the most common complications is associated with object and process presence detection and representation, and presence-based action, reaction, and interaction. Presence is often associated with first-person user experience in virtual environments [3]. In this paper, we define presence as the availability of an entity for various actions of, reactions by, and interactions with a system, and as a potential modifier of structural, functional, or behavioral system dynamics. Thus, presence is relative and assessed for any external object (not only users), and by any internal agent. Presence is conceived from the point of view of a system, a subsystem, an agent, or even an elementary component – especially in complex, distributed and cyber-physical systems. The notion of object *presence* and process or event *occurrence* is critical for realistic modeling of complex, unreliable settings, scenarios, and situations.

Consider, for instance, the issue of sensitivity to light. Many technologies and applications are adjusted to varying light conditions. These systems have to detect and monitor

presence or absence of light in order to activate or deactivate various illumination-dependent capabilities, such as infra-red vision, image processing algorithms, display brightness, and palette adjustment. Therefore, capturing the presence or absence of light in the model is critical for correct understanding and realization of processes in the system. Consider what it would mean to apply presence-aware modeling to gravity, for instance. These examples can be easily generalized to a wide variety of surrounding conditions affecting internal dynamics.

External object/process detection is a substantial field of research and development in various domains, especially autonomous systems with visual or sensory object detection capabilities [4], [5] and cyber-security [6], [7]. General-purpose modeling frameworks have also been proposed [8]. Nevertheless, integrating presence—absence analysis into system and process models is still rare, unless it related to the product's core function, e.g., smoke detector. Countless system models provide no support for presence problems beyond the trivial and explicitly-specified ones. While various interactions can rely on simplifying presence-related assumptions, systems are becoming ever more complex, and have to act in various exceptional conditions and configurations.

Naïve models take the presence of external objects and occurrence of external processes for granted. In contrast, enriched presence-aware models explicitly tackle this issue as inherent in the system's conceptual design. Defining entities in a model does not guarantee their presence in the environment of the modeled system. Likewise, refraining from defining entities in the model does not render them absent in the same context. The naïve modeling approach ignores or relaxes these complications of reality, creating a glaring gap between the model and the real system. The naïve model can help understand the "big picture", but must then be elaborated into a realistic, high-fidelity model. Otherwise, it will likely be abandoned due to failure to provide interesting insights to the system designers.

With this background in mind, we propose to enhance the model-based systems engineering paradigm in order to account for presence or absence of environmental objects in dynamic, unreliable settings. The underlying paradigm shift fosters modeling notation and capability that naturally support such varying scenario modeling. The models' capability to support both mainstream and variant profiles is rather straightforward in Object-Process Methodology (OPM) [9], a conceptual modeling framework for complex systems. Our approach capitalizes on insights from conceptual modeling of exceptions and dynamics of complex biochemical systems, which have been studied and analyzed with OPM [10]–[12]. In these models, the existence of various materials and catalysts at sufficient concentrations, occurrence of various molecular interactions, and satisfaction of environmental conditions (temperature, light, humidity, etc.) are all fundamental for the result of the biochemical reaction. A rigorous model cannot neglect these aspects in describing the functional results or outcomes in such settings. Moreover, the capability to simulate these various outcomes depends on the correct modeling of the presence of surrounding conditions and the process's dependency on and sensitivity to each presence configuration.

The rest of this paper is organized as follows: Section II provides a brief description of OPM and its benefits in tackling presence-aware modeling. Section III introduces several techniques to enrich system models with presence-awareness, while preserving the structure, behavior, and notation of the core model. In Section IV we discuss the conclusions of this study, summarize the paper, and outline future research.

II. Object-Process Methodology

Object–Process Methodology (OPM) [9], our underlying modeling framework, is a holistic conceptual modeling methodology for the design and analysis of complex systems and processes. It has several important strengths making it a useful and attractive modeling framework in various domains, including complex socio-technical systems, aerospace and defense, information systems, medicine, biochemistry, embedded software, and space exploration. Unlike most conventional modeling frameworks, OPM is domain-independent, and has an intentionally unified, integrated view of form, function, and behavior. OPM's free CASE tool, OPCAT[1] [13] provides robust simulation capabilities. OPM's static-dynamic and structural-procedural unified view has proven complexity management and alleviation capabilities that cater to complex system and process modeling [14]. OPCAT's simulation mechanism was shown to enhance understanding of behavioral and procedural aspects, which are otherwise very difficult to perceive in static visual modeling frameworks and models [15]. OPM is an emerging ISO standard (ISO 19450) and the basis for a new generation of model-based ISO standards. It is a notable, leading modeling and design methodology [16], and as a leading conceptual modeling framework [17].

OPM is both graphical and textual: each model notion captured visually in an OPM diagram (OPD) is accompanied by an automatically generated formal textual description in Object-Process Language (OPL). OPM's modeling elements are **Process** (ellipse), **Object** (rectangle), and **state** (*rountangle*). These elements can be connected by structural and procedural **Links**, capturing various types of relations. An OPM metamodel describing objects, processes, and structural links is shown in Fig. 1. Procedural links and states are shown in Fig. 2. The OPL sentences were copied into the diagram on top of their respective visualizations (they are not part of the original diagram). OPDs are hierarchically organized, interdependent, and self-similar. Each OPD can be extended to lower levels through unfolding, optimized for structural and functional decomposition, in-zooming, for procedural and behavioral elaboration, or view derivation, used for elaboration of model aspects around pivotal model elements.

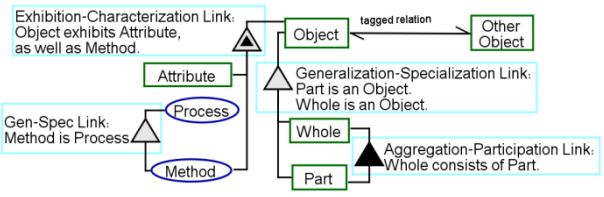

Fig. 1. OPM Notation (a) – Objects, Processes, and Structural Relations

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

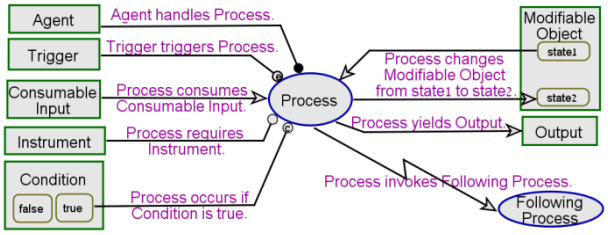

Fig. 2. OPM Notation (b) – Procedural Relations and State Dynamics.

III. PRESENCE-AWARE MODELING WITH OPM

This section describes several system modeling extension techniques for capturing presence-related exceptions and variants while preserving a unified model, and linkage between the flow of control and branches specifying irregularities. OPM has been applied for conceptual modeling in various domains including non-traditional system modeling domains such as biochemical and biogenetical systems [11], [18]. A metamodel and notation for exception modeling has also been developed [10], [19]. These models propose several important modeling extensions and enhancements for exceptional systems settings.

A. Presence-Sensitive Modeling

A naïve modeling pattern, in which presence is ignored, is illustrated in Fig. 3. The diagram captures typical relations between a process and some role-carrying objects: **Enabler**, **Disabler**, **Consumee**, **Affectee**, and **Resultee**. This model implicitly assumes that **Enabler**, **Affectee**, and **Consumee** are initially **present** in the setting, while **Disabler** and **Resultee** are **absent**. An extended, presence-aware model is illustrated in Fig. 4. It captures presence—absence using presence-reflective object states. It is enriched with notions of object presence—absence conditionality for the same roles in the naïve model. The two approaches are compared in TABLE I. TABLE I.

The use of states may be confusing when states are required to characterize objects for functional purposes. In order to distinguish presence profiling from functional profiling, we discern two different **Object** attributes: **Presence** and **State**, as illustrated in Fig. 5. Processes are characterized by **Occurrence** rather than **Presence**. OPM notation supports linking states of one object (or values of one attribute) with another for enhanced verification and control. In Fig. 5, for instance, **Object** can only be in **state1** or **state2** when it is **present**, and in **state3** when it is **absent**. Occurrence can be either **potential**, **ongoing**, or **over**, an important distinction for pre-, per- and post-occurrence dynamics.

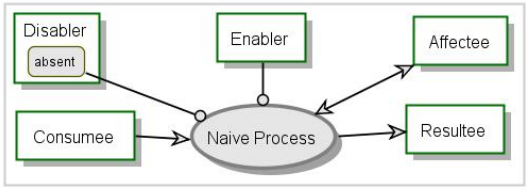

Fig. 3. A naïve OPM system model

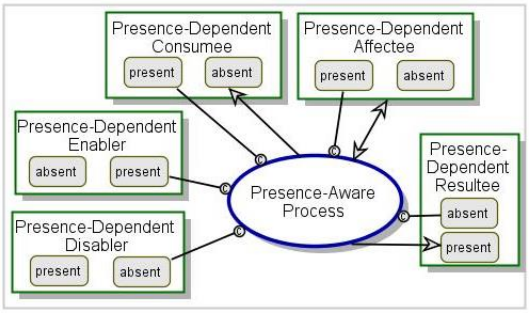

Fig. 4. Presence-aware OPM model

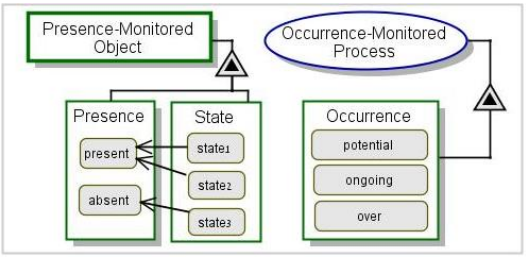

Fig. 5. Presence-monitored object and process characterization and relations

B. Gene Configuration Modeling

Consider, for instance, the military active protection system (APS) for armored vehicles, Trophy [20]. It can supposedly detect, classify, track, and destroy incoming missiles and rockets. The system consists of a small mobile scanner, a command and control computer, and a set of countermeasures, such as a shotgun-like blast, decoy dispersion, and electromagnetic pulse. When operational, we can assume that

TABLE I. COMPARISON OF NAÏVE AND PRESENCE-AWARE MODELS

Action	Model Implications	
	Naïve	***Presence-Aware***
Enabling	**Enabler** enables process execution regardless of its presence.	**Presence-dependent enabler** allows the occurrence of the process only if it is **present.**
Disabling	**Disabler** must assume the state **absent** in order to enable the process. OPM basic notation does not support a disabling relation; hence a state-independent disabler cannot be supported.	**Presence-dependent disabler** prevents the occurrence when it is present, which means that it allows the occurrence when it is **absent.**
Consuming	**Consumee** is consumed by the process if it exists. The process will not run if the **consumee** does not exist by default or after an earlier process has created it.	**Presence-dependent consumee** is not consumed, but marked as absent, only if it is **present.**
Creating	**Resultee** is created, or re-created, regardless of its previous existence.	**Presence-dependent resultee** is generated as **present**, only if it was first **absent**
Affecting	**Affectee** will be modified by the process if it is present, otherwise an execution error will occur.	**Presence-dependent affectee** is modified only if it is **present**. The states it exits and enters are not presence-related (this would be equivalent to consume/ resultee).

the system runs a cyclical process. Each activity takes place according to the availability, presence, or occurrence of some key behavioral modifiers, e.g., a general threat, a specific threat, surrounding friend or foe vehicles, available countermeasures, etc. This is a configurable, modifier-presence-dependent process.

Gene Configuration Modeling is a design pattern whereby essential system setting definers are encoded as the system's "genes". This technique is inspired by the evolving conceptual model of RNA-related processes [11], [12], [18]. The presence or absence of genes in the setting modifies the system's overall structural and behavioral configuration. The set of system genes and their assigned presence statuses is therefore called **Gene Configuration**. A gene configuration model allows for the generation of various system configurations. The gene configuration pattern is obtained by applying presence-sensitivity to the system's elementary configuration variables, which are typed or characterized as Genes.

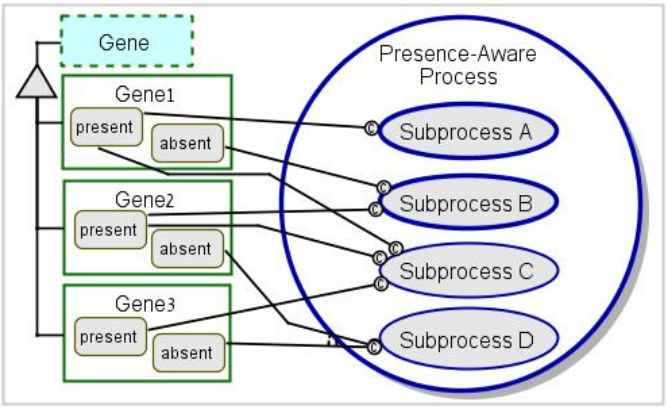

Fig. 6. Gene Configuration Model

A gene configuration metamodel is illustrated in Fig. 6, showing an example with three genes and four subprocesses. Each subprocess occurs based on the presence or absence of the various genes. OPCAT automatically formulates the gene configuration required for the occurrence of each subprocess:

```
Subprocess A occurs if Gene1 is present.
Subprocess B occurs if Gene1 is absent and Gene2 is present.
Subprocess C occurs if Gene2 is present, Gene3 is present, and
Gene1 is present.
Subprocess D occurs if Gene2 is absent or Gene3 is absent.
```

C. Representation Gaps in Presence-Aware Models

Physical-Informatical Essence Duality (PIED) is the notion that any entity exists once as the real-world (usually physical) embodiment, and once as a perceptional representation, obtained and maintained by the systemic agent interacting with the external entity [21], [22]. Models must account for this notion, and distinguish the real-world, external entity, from its internal representation. Interactions between the system and the entity are based on the representation held by the system, and when the representation's characteristics do not match the real-world entity's characteristics, problems and failures may occur.

Presence-aware models can be further enriched by adopting a PIED-aware approach. When the process depends on the presence or absence of an external entity, the system must detect and acknowledge the entity's presence status. The entity may be present in the scene, but not detected by the system. Alternatively, it can be absent but thought to be present. When presence detection is incoherent, processes may not occur when they should, occur when they should not, or lead to erroneous, often undesired results. Hence, presence and absence detection must be part of the model, with the interaction depending on the representation of the external entity's presence attribute – not on the actual presence.

Consider a gas leakage neutralization mechanism, which has to detect the presence of a particular gas in a monitored volume, and halt the flow in case of extreme gas levels. The gas itself may be hazardous or merely indicative of another process taking place, or not. If the mechanism fails to detect gas presence in the volume, no leakage stopping action will occur. A malfunctioning or non-calibrated monitoring device might miss the leakage. When finally detected, it may be too late to interfere. In an alternative scenario, a process has to be triggered based on gas absence, e.g., in vacuum conditions. If the absence of air is not detected, the process will never begin.

PIED characterizes external genes whose correct internal representation is uncertain and unascertainable. In Fig. 7, **subprocess B** from Fig. 6 is in-zoomed (without loss of generality) in a presence-PIED-aware manner. The relevant genes are **Gene1** and **Gene2**. Each gene has to be **present** in order to induce **Stimulating**, which invokes its dedicated **Gene# detecting and representing** process. In turn, a ***representation*** **Gene#.Rep** is generated and set to **as-present**, indicating its representativeness. Representations are **as-absent** by default, and remain this way as long as no stimulation and detection take place. The conditional links enabling or disabling the subprocess in Fig. 6 are substituted by conditional links between the **Gene#.Rep**'s presence-indicative values and **function**. We assume, but this is not mandatory, that stimulating always invokes detecting, and that detecting and representing are always correct.

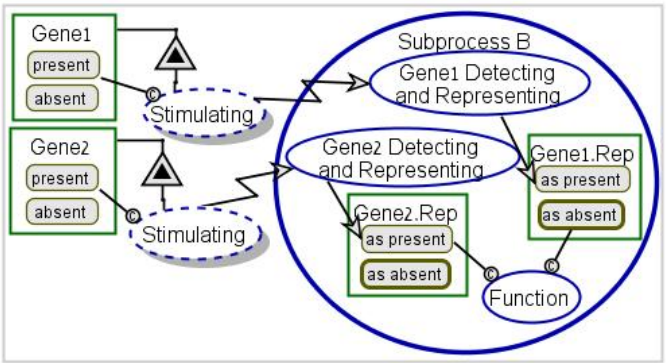

Fig. 7. PIED-aware presence-sensitive process

D. Presence-Aware Risk Modeling

System capability to overcome and adapt to presence issues of various assets, facilitators, and functionalities, is part of its overall robustness. Robust systems are capable of operating even when the environmental setting of operation is imperfect or incomplete. Robustness is originally defined as an inherent property of evolving organisms, so robustness and evolvability are analogous, intertwined, and almost synonymous [23].

A framework for enhancing conceptual system models in order to improve and facilitate robustness, flexibility, and resilience, and thereby reduce technical and programmatic risk, has been specified [24]. The framework demonstrates the extension of nominal models to account for exceptional risk-posing settings. Presence is associated with risk in a dual manner: on the one hand, risk is amplified by the presence or absence of risk-posing agents and contained by the complementary. Conversely, risk is amplified by the absence of risk-countering agents, and contained by their presence. Capturing the presence and absence of systemic and external agents in the same conceptual model enables modeling and simulation of corresponding multiple an interchangeable functionalities, structural design alternatives, and adaptive configurations, increasing the system's robustness.

A presence-aware model also makes it possible to study various configurations of risk sources and risk mitigation agents, and to conduct "what-if" analysis on top of the core system model, based on multiple scenarios. Thus, the design becomes risk-oriented as well, i.e., risk consideration direct the design and affect it. Risk-oriented design is not about *probability* but about *possibility*. It is more important to capture the latter than the former in order to devise means to handle risk. For instance, the absence of some critical component or functionality, perhaps due to various constraints and design considerations, can lead to severe outcomes unless mitigated. These effects must be modeled regardless of their probability. Without presence-aware modeling, those *phantom* elements cannot be represented in the model, and the implications of their absence from the system cannot be demonstrated or evaluated on top of the operational model.

The basic risk modeling pattern is illustrated in OPM in Fig. 8. This model captures **Risk** as characterized by a **Risk Source**, which is an ***object***, and a **Risk Effect**, which is a ***process***. The **Risk Source** triggers the **Risk Effect** when **Risk Source** becomes **risk-posing**. The **Risk Effect** affects an **Asset**, changing its **Asset State** from **ok** to **inflicted**. The occurrence of **Risk Effect** invokes a **Risk Monitoring** service, which in turn invokes a **Risk Responding** activity. The **Risk Responding** activity is supposed to restore the **Asset State** to ok but it can also partially succeed and regain only a salvaged **Asset**. The basic model is indifferent to presence and essence duality. System and risk analysts might encounter challenges, such as risk source detection and representation, presence as a reason for risk posing, system/model awareness to risk source presence, risk effect occurrence detection, and risk responding result acquisition. The latter is especially critical when the risk handling agent and the asset are loosely coordinated.

In order to meet the presence-sensitive risk modeling challenges above, we extend the basic naïve model. The extended model is illustrated in Fig. 9. **Risk Source** and **Risk Effect** are marked as external entities. **Risk Source** activates **Risk Effect** when it is absent in the scene – as long as it is present no risk will materialize. The **Risk Effect** itself is characterized by an **Occurrence** attribute, which is first changed from **potential** to **ongoing**, and later from **ongoing** to **over**. In addition, the **Risk Effect** creates a **Risk Effect Impact** random variable, which indicates whether the effect is **detected** or **undetected**. Only when **Risk Effect** is **detected** can the **Risk Monitoring** process commence, and it then updates **Risk Status**, the internal representation of the risk. **Risk Responding** is triggered when **Risk Status** is either **occurring** or **over**, and we can assume that the results of the response activity will be affected by the time elapsing from detection to response. While ongoing threats can be handled immediately, the impact of realized and terminated ones can be severer and much more difficult to repair.

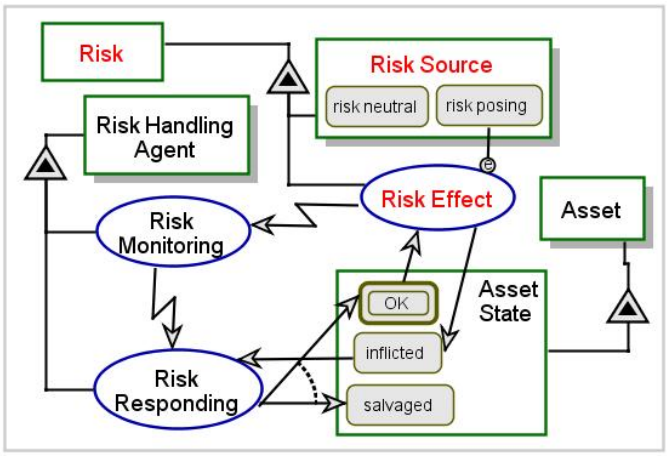

Fig. 8. A basic risk modeling pattern

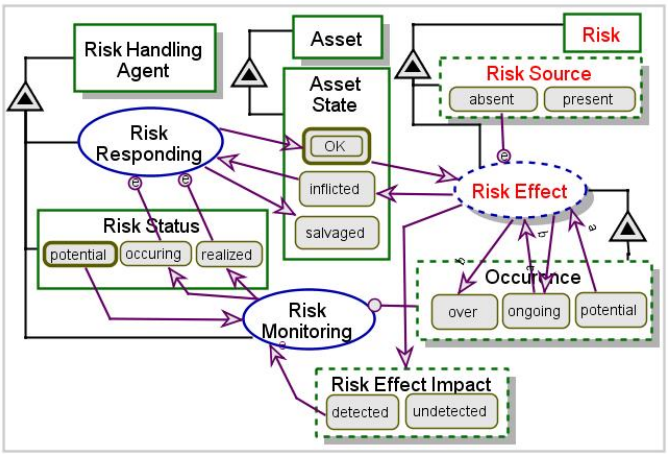

Fig. 9. A presence-sensitive risk modeling pattern

The benefits from this extended model include: (a) defining presence and occurrence in their different combinations as functional triggers, (b) capturing and understanding the meaning of incomplete or wrong information in risk mitigation, and c) simulation of various scenarios and anomalies which might prevent the system from functioning appropriately. This model attempts to capture both robust design and risk-oriented design in the same view. The presence or absence of various state elements creates permutations of states. This requires several alternative process variants, supporting some of the exceptional configurations. Some critical risk-posing agents' presence—absence can lead to failures and undesired results, and should be addressed by adequate risk mitigation.

IV. SUMMARY

Norbert Wiener laid the foundations for the mutual adoption of conceptual models in biological systems by technical systems and vice versa [25]. This paper follows the Wienerian approach and incorporates conceptual biological modeling notions and aspects into generic, complex cyber-physical systems. We have focused on the concept of presence and presence-awareness in system models. Intelligent and primitive organisms alike have the natural awareness and sensitivity to the presence—absence of external objects and processes, and for presence-based reaction, action, and interaction. Modeling paradigms are often naïve, deterministic, and focused on nominal common wisdom modeling. We emphasize the importance of exception-aware, oriented, and integrated modeling. Nominal scenarios and configurations constitute the ultimate majority, but most of the design and development effort is, or should be, spent on exceptional, anomalous aspects, despite their small, often extremely marginal proportion. Presence exceptions are a common challenge, especially when coupled with presence-dependent system dynamics. Unified, integrated modeling of nominal and exceptional settings in the same model facilitates presence-awareness and sensitivity. This approach is made possible with careful utilization of Object-Process Methodology, which has also been shown to cater to biological systems analysts.

In this paper, we have described several techniques for enriching and extending system models with presence-related exceptions. The basic presence-aware modeling notation, semantics, and design pattern enhance naïve models with presence-awareness and presence-sensitivity. The "gene configuration" concept facilitates system configuration modeling and control based on multiple fundamental structure, function, and behavior definers. Integrating presence-awareness with physical-informatical duality provides for presence perception and representation gaps within the system. Finally, we have shown how risk-oriented models can benefit from presence-awareness by capturing those risk-sources whose potential to create risk is derived from their undesired absence or presence. Future research includes application of the various presence-aware and presence-sensitive modeling techniques to actual problems. In addition, we intend to analyze the results of simulated presence-aware OPM models and utilize the simulation for gene configuration permutation set coverage and verification.

ACKNOWLEDGMENT

This research was funded by the European Union's 7th Framework Programme (FP7/2007-2013) under grant agreement No. 262044 – VISIONAIR and carried out the Enterprise Systems Modeling Laboratory at the Technion.

REFERENCES

[1] A. Rosenblueth and N. Wiener, "The role of models in science," *Philos. Sci.*, vol. 12, no. 4, pp. 316–321, 1945.

[2] Y. Y. Haimes, *Risk Modeling, Assessment, and Management*, 3rd ed. John Wiley & Sons, 2009.

[3] K. M. Lee, "Presence, Explicated," *Commun. Theory*, vol. 14, no. 1, pp. 27–50, Feb. 2004.

[4] E. Pedersen and T. Sokoler, "AROMA: abstract representation of presence supporting mutual awareness," *Proc. ACM SIGCHI Conf. ...*, pp. 1–13, 1997.

[5] J. Steffens, E. Elagin, H. Neven, E. Interfaces, and S. Monica, "PersonSpotter - Fast and Robust System for Human Detection , Tracking and Recognition."

[6] B. S. J. Scott, "Threat Management Systems The State of Intrusion Detection," 2002.

[7] G. J. Victor, M. S. Rao, and V. C. Venkaiah, "Intrusion Detection Systems - Analysis and Containment of False Positives Alerts," *Int. J. Comput. Appl.*, vol. 5, no. 8, pp. 27–33, 2010.

[8] H. Christein and P. Schulthess, "A general purpose model for presence awareness," *Distrib. Communities Web*, 2002.

[9] D. Dori, *Object-Process Methodology: A Holistic Systems Approach*. Berlin, Heidelberg, New York: Springer, 2002.

[10] Y. Somekh, M. Peleg, and D. Dori, "Classifying and modeling exceptions through object process methodology," in *2007 International Conference on Systems Engineering and Modeling - ICSEM'07*, 2007.

[11] J. Somekh, M. Choder, and D. Dori, "Conceptual Model-based Systems Biology: mapping knowledge and discovering gaps in the mRNA transcription cycle.," *PLoS One*, vol. 7, no. 12, p. e51430, Dec. 2012.

[12] J. Somekh, G. Haimovich, A. Guterman, and D. Dori, "Conceptual Model-based Systems Biology□: Advancing Knowledge of the mRNA Decay Process via Combining in silico with Wet Laboratory Experimentation," pp. 1–21.

[13] D. Dori, C. Linchevski, and R. Manor, "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.

[15] Y. Yaroker, V. Perelman, and D. Dori, "An OPM conceptual model-based executable simulation environment: Implementation and evaluation," *Syst. Eng.*, vol. 16, no. 4, pp. 381–390, 2013.

[16] J. Estefan, "Survey of model-based systems engineering (MBSE) methodologies," 2007.

[17] D. Dori, "Object-Process Methodology for Structure-Behavior Codesign," in *Handbook of Conceptual Modeling*, D. W. Embley and B. Thalheim, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 209–258.

[18] D. Dori and M. Choder, "Conceptual modeling in systems biology fosters empirical findings: the mRNA lifecycle.," *PLoS One*, vol. 2, no. 9, p. e872, Jan. 2007.

[19] M. Peleg, J. Somekh, and D. Dori, "A methodology for eliciting and modeling exceptions.," *J. Biomed. Inform.*, vol. 42, no. 4, pp. 736–47, Aug. 2009.

[20] R. Hughes, "Israel Armor Protection System 'Revolutionary,'" *Jane's Defense Weekly*, pp. 1–3, 2005.

[21] Y. Mordecai, C. Chapman, and D. Dori, "Conceptual Modeling Semantics for the Physical-Informatical Essence Duality Problem," in *IEEE International Conference on Systems, Man, and Cybernetics - SMC2013*, 2013.

[22] Y. Mordecai, O. Orhof, and D. Dori, "Modeling Software Agent Awareness of Physical-Informatical Essence Duality," in *IEEE International Conference of Software Science, Technology, and Engineering - SwSTE 2014*, 2014.

[23] H. Kitano, "Biological robustness.," *Nat. Rev. Genet.*, vol. 5, no. 11, pp. 826–37, Nov. 2004.

[24] Y. Mordecai and D. Dori, "Model-based risk-oriented robust systems design with object-process methodology," *Int. J. Strateg. Eng. Asset Manag.*, vol. 1, no. 4, pp. 331–354, 2013.

[25] N. Wiener, *Cybernetics, or control and communication in the animal and the machine*, 2nd editio., vol. 2, no. 5. New York, London: MIT Press and John Wiley & Sons, Inc., 1948.