





# Creon: A Capability Reference Ontology for the Analysis and Design of Systems Capabilities

Rodrigo F. Calhau<sup>1,2,3</sup> , João Paulo A. Almeida<sup>2</sup> ,  
Raquel Hoffmann<sup>3</sup> , and Giancarlo Guizzardi<sup>4</sup> 

<sup>1</sup> LEDS Research Group, Federal Institute of Espírito Santo, Brazil  
calhau@ifes.edu.br

<sup>2</sup> Ontology & Conceptual Modeling Reserach Group (NEMO),  
Federal University of Espírito Santo, Brazil  
jpalmeida@ieee.org

<sup>3</sup> System Design and Management Graduate School, Keio University, Japan  
raquel.hoffmann@keio.jp

<sup>4</sup> Semantics, Cybersecurity & Services, University of Twente, The Netherlands  
g.guizzardi@utwente.nl

**Abstract.** Enabling system-wide capabilities is a key aspiration of systems engineering. It comes as no surprise then that a number of system modeling techniques have turned their attention to the analysis and design of capabilities. Despite that, these tasks have remained challenging due to the interactive and emergent nature of capabilities, as evidenced by the cases of agility, resilience, adaptability and security. Modeling capabilities requires a proper understanding of capability-related concepts and also how capabilities interact and emerge. To address these challenges, this paper proposes a capability reference ontology (*Creon*) grounded in system science, emergence explanations, and disposition theories. The ontology aims to enhance the modeling of emergent capabilities and capability relationships in systems engineering. By leveraging concepts from the Unified Foundational Ontology (UFO), *Creon* supports a comprehensive representation of capabilities. We illustrate the use of the ontology in a transportation context, and present some implications for system capability modeling.

**Keywords:** capabilities · dispositions · emergence · ontologies · systems modeling

## 1 Introduction

Providing desired system-wide capabilities while avoiding other undesirable properties, such as vulnerabilities, is a primary objective of system engineering (i.e., developing, designing, maintaining, or creating systems). To facilitate the understanding of system capabilities, they have been modeled in different ways in different fields. E.g., system (functional and non-functional) capabilities have been modeled directly or indirectly through descriptions of system requirements, often employing notations such as UML [47] and SysML [54] (e.g., using requirements and use case diagrams) or notations from the Enterprise Architecture (EA) area, such as ArchiMate [58] and Unified Architecture Framework (UAF) [35] (which have explicit ‘capability’ constructs).

While these notations are used to articulate descriptions of ‘capabilities’ in different ways, they lack an explicit foundation for capabilities. This is especially the case for how capabilities interact and how they emerge or are implemented. E.g., system-wide capabilities like security, resilience, and scalability are the result of complicated combinations of elements. A resilient system does not always have resilient components, be they hardware or software. In this case, resilience can result from flexible and adaptable components. Hence, modeling these system-wide capabilities requires a foundation that can account for how they emerge from the capabilities of the various system components.

Such a foundation can be provided by establishing the semantics of language constructs in terms of a well-founded ontology. Ontologies can improve conceptual modeling’s expressiveness and domain appropriateness in terms of representing system aspects<sup>5</sup>. Moreover, they have proven helpful in the computing field, according to [21], for formalizing the semantics of different kinds of artifacts. Many ontologies have been proposed to explain system aspects in distinct fields, including, but not limited to, business capabilities [5,49], competences [14], vulnerabilities [46,49], and system functions and qualities [23]. Ontologies about capabilities of different types of systems have been proposed, such as systems-of-systems [18], enterprise systems [44], engineered systems [4], cyber-physical systems (CPS) [16,24,60], manufacturing systems [25,28,51]; among others [31]. Nevertheless, they tend to have a narrow understanding of capability concepts and concentrate on addressing technological and practical problems associated to a particular domain, thus neglecting other fundamental aspects such as dealing with the phenomena of capability interaction and emergence.

We attempt to fill this gap in this paper by using notions from system science [1,8,9,11], emergence explanations [27,40,45,55], and disposition theories [7,19,38,41]. This effort is complementary to our early work [13], in which we approached the structural aspects of systems that make emergence possible (components, connections, and constraints), but in which we did not approach capabilities and their relationships explicitly. It is also complementary to [12], where we addressed the emergence of organizational capabilities from personal competences grounded on disposition theories (e.g., [7,19]), but in which we did not propose a general ontology for capabilities and their context. In this paper, we focus on such a reference ontology. Our proposed reference model is based on the Unified Foundational Ontology (UFO) and represented using the OntoUML modeling language [21]. We follow the SABiO approach [2] and assess the ontology by showing how it address the competency questions, as well as illustrating its instantiation in a number of cases, including its use in modeling a particular organizational domain. We discuss the implications of the proposed ontology to the conceptual modeling of capabilities. The remainder of this paper is structured as follows: Section 2 presents an overview of the literature related to capabilities and UFO; Section 3 presents the main result of this work, the Capability Reference Ontology (*Creon*); Section 4 discuss some implications of the ontology to capability representation; Section 5 discusses related work, and Section 6 concludes with our final remarks.

---

<sup>5</sup> Following [21], by ‘ontology’ we do not mean a description logic specification but a theory articulating domain distinctions as a result of an ontological analysis process.

## 2 Background

In this section, we address some relevant capability-related concepts. We also address the notion of disposition, which is key to our account of capabilities. In the end, we also briefly review the foundational base we use to ground *Creon*.

### 2.1 Capabilities

Generally speaking, capabilities are abilities, with the a widely accepted definition referring to capability as an ability to accomplish a certain outcome [39]. Ability, in its turn, is often described as a propensity of behavior or as a type of dispositional aspect that permits an individual (usually an agent) to perform an action, whether or not it has value [32]. Maier [32] defines the notion of ability more specifically as the “power that relates an agent to an action” [32]. Ability, then, is a potentiality associated with both an action and a bearer. In light of this definition, what sets abilities apart from capabilities? The value factor, or the intended outcomes, is one of the primary differences. Ability descriptions are typically more general and do not state the desirable performance-related outcomes, results, or accomplishments. On the other hand, capabilities have a teleological component that is connected to an individual’s purpose [31].

E.g, in EA and systems engineering areas, capabilities are defined as the “ability to do something *useful*” (our emphasis) [33, 34, 37]; in IS as “ability to achieve a *desired* effect” (our emphasis); and, in the military field, it is defined as “ability to achieve a *determined military objective*” (again, our emphasis) [3]; As illustrated, a teleological aspect is present in capability definitions in distinct areas. Many of these definitions were studied by Tell [56], who generalized a definition for capability. The author defines a capability as the possibility of achieving specific results with regard to specific source entities and stakeholders.

Martin et al. [36] conducted a systematic literature review on the concept of capability in the field of system-of-systems and EA. A total of 77 works were considered in the research, aimed at gaining a better understanding of the definitions in the area as well as frameworks and guidelines. As the authors detail, most of the works come from the defense sector and define capability essentially as having the same meaning presented above, i.e., as the “power or ability to achieve some result”. Based on these definitions, a set of keywords was identified and classified into four groups: (i) ability, power, and capacity; (ii) effects; (iii) patterns, and conditions; and (iv) task, mission, function, and action. In a similar way, Loucopoulos & Kavakli [31] states that capabilities integrate four key perspectives within an enterprise: (i) *teleological* perspective, which justifies the existence of capabilities; (ii) *operational* perspective, which deals with the use and implementation of capabilities; (iii) *service* perspective, which connects capabilities to the value they deliver to stakeholders; and (iv) *context* perspective, which ties capabilities to their specific situation and environment.

Martin et al. [36] also identifies common characteristics to the definitions of capabilities, such as: (i) *composition*: capabilities have sub-capabilities; (ii) *abstract nature*: capabilities describe “what” (but some can include “how”); (iii) *boundaries*: defined through measurement and desired effects; (iv) *measurement*: capabilities can have performance metrics, i.e., they can be gradable; (v) *directional view*: capabilities can be

seen vertically (functional area) or horizontally (cross-cutting functions); (vi) *evolution*: capabilities can evolve regarding time; (vii) *engineering*: capabilities can be engineered through iterative, incremental, feedback-based processes.

**Capabilities as Dispositions** We adopt the notion of *disposition* to account for capabilities. This notion has been the subject of study in the ontology and philosophical literature, with applications in various fields. Capabilities can be viewed as dispositions of a special type, which can bring insight into how to conceptualize and represent them. Dispositions typically encompass characteristics such as propensities, affordances, capacities, potentials, tendencies, and powers. These characteristics are generally similar in that they give their bearers the possibility to exhibit a particular behavior or produce a particular outcome in specific circumstances. While specific circumstances may be required for dispositions to manifest, they may also fail to do so even in the right circumstances. When they manifest, it is because they are triggered (activated) by a situation (context). As a result, they are manifested in terms of the occurrence of certain events [38, 41]. A classical example is the case of the magnet's disposition to attract ferrous objects. When a magnet  $o_1$  is close to one such object  $o_2$  (situation/circumstance), the attraction disposition  $d_1$  of  $o_1$  is manifested together with the disposition  $d_2$  of being attracted within  $o_2$ . In any case, even if the disposition  $d_1$  is manifested or has never manifested – e.g., in case  $o_1$  has never been near a ferrous object – the magnet  $o_1$  has this disposition. In other words,  $o_1$  possesses  $d_1$ , independent of  $d_1$ 's manifestations or lack thereof. Dispositions and their manifestation through events are strongly associated [7, 19]. Complex actions can be interpreted as the outcome of the “combination” of various dispositions. This includes interactions as *triggering*, *preventing*, *complementarily activating*, and *mutually activating* [7, 19]. As explained in [41], dispositions can be “combined” to form new dispositions, e.g., by adding dispositions together (yield a new added resultant disposition) or by subtracting them (reducing the resultant disposition). Dispositions cannot always be combined as a simple sum (using “additive composition”). In such cases, it is necessary to have a systemic, “nonlinear” understanding of these dispositions and their interactions (ibid.).

**Capability and Emergence** Emergence in systems science is the phenomenon where a system exhibits characteristics or actions that its constituent parts would not possess on their own [9]. Emergent powers (dispositions) must be associated with a certain uniqueness [41]. In addition, they need to be independent with respect to the “emergent base” [42]. Bunge [9, 10] divides aspects of the system into emergent and resultant. Aspects of a system that can be broken down, clarified, or reduced to aspects of its constituent parts are known as *resultant aspects*. For instance, the simple sum of the weight of a system's constituent parts determines the system's overall weight. Emergent aspects, as opposed to resultant aspects, do not exist by themselves in the disconnected components. E.g., the flotation capacity of a ship cannot be reduced to flotation capacities of its constituent parts (by itself, a random section of a steel hull typically cannot float). The emergent aspects supervene on the aspects of the parts [45]. Some authors [27, 40] add that emergent aspects are also the outcome of system restrictions, which both restrict them and allow for the emergence of novel traits. Juarrero [27] provides an example of this by showing how the knee joint limits the movements of the femur and tibia, which

helps the walking ability to develop. Similarly, emergent aspects are a direct result of the connections between the components [55]. E.g., the manner in which molecules of carbon are linked is what separates diamond from graphite. These variations give rise to unique emergent characteristics, such as the conductivity of electricity in graphite and transparency in diamonds.

## 2.2 Ontological Baseline

Here, we use a portion of the UFO foundational ontology [20], which defines a system of domain-independent categories and their ties, to more accurately account for capability-related phenomena. Conceptualizations of phenomena of interest can be expressed using the domain-independent categories and their relationships defined by UFO. Theories from Formal Ontology, Philosophical Logics, Philosophy of Language, Linguistics, and Cognitive Psychology have all been incorporated into the development of UFO [21]. The used fragment encompasses the first two fundamental ontological categories: those of *individuals* (specific things, such as John, Mary, Saturn, and The Beatles) and those of *types* (concepts, universals, such as Person, Planet, and Music Band). There are two types of individuals: concrete and abstract. Concrete individuals are divided into *situations*, *endurants*, and *events* (also known as *perdurants*).

Events are things that take place in time, such as tasks, activities, and processes. Events can be atomic or complex and are causally related. In addition, endurants, including their aspects, can be created, altered, or terminated by events [22]. *Endurants* are individuals that endure over time while maintaining their identity (i.e., people, organizations, cars, contracts). Endurants are classified into *objects* and *aspects*. An endurant that is regarded as existentially independent is called an *object* (e.g., John's car). *Functional complexes* are objects formed by components that carry out different functional roles with respect to the whole. An *aspect* is a reified characteristic that is existentially dependent on another endurant (referred to as its bearer). Aspects have their own lifecycle and can be created, destroyed, or undergo other qualitative changes over time (as much as objects).

We are particularly interested in the UFO concept of *dispositions*. Aspects known as *dispositions* can be made observable by the occurrence of *events* (perhaps the actions of agents, like Anna speaking English). Dispositions are said to be "activated" by *situations* where they might manifest, such as when a magnet is near ferrous material, or when Anna is requested to bring up the subject of a meeting. As endurants, they are able to experience changes over time and undergo qualitative transformations [20]. Situations circumscribe a portion of the world that can be understood as a whole, with objects exhibiting certain properties and standing in certain relations (e.g., John being two meters apart from Mary; Mary being 1.8 meters tall; John being married to Mary; Mary having Dengue Fever; John having a Heart that once belonged to Paul).

We are modeling our reference ontology using OntoUML as a language that incorporates the UFO distinctions and axiomatization [21], thus supporting the construction of models that are ontologically well-founded according to UFO. From a syntactical point of view, OntoUML is a UML profile that uses stereotypes (and colors) to incorporate the fundamental distinctions put forth by UFO. There are a number of UFO-compliant core ontologies created with OntoUML. One that is particularly relevant for this work

is the Disposition Interaction and Emergence Core Ontology (Disc-O).<sup>6</sup> This ontology uses several theories of dispositions and emergence to make explicit how dispositions interact and emerge, including relations of reciprocity [19,41], complementarity [41], enabling and disabling [7], changing [41], among others.

### 3 The Capability Reference Ontology (*Creon*)

This section presents the proposed capability ontology. This ontology was built following the Systematic Approach for Building Ontologies (SABiO) [2]. This approach starts by identifying the main purpose of the ontology, which is to support the modeling of capabilities, particularly the modeling of capability interaction. For this purpose, a number of requirements were elaborated. In summary, the ontology must consider: (i) the concept of capability (and related concepts) in a general sense; (ii) the different relations between capabilities; (iii) the different types of capabilities. Based on these requirements, the main competency questions (CQ)<sup>7</sup> were formulated, including: What differentiates capabilities from dispositions of other types? How can capabilities be classified? How can they be interrelated? How do interrelated capabilities manifest? How do capabilities emerge?<sup>8</sup>

Based on this, the key concepts related to each ontology were identified: capability, capable object, capability context, capability manifestation, capability output, capability outcome, capability type, capability phase, and capability-bound role. In addition to the main concepts, the ontology engineering approach also outlines the key relationships between them, such as the interaction between capabilities (based on disposition theories), emergence relationships, and part-whole relationships. Finally, the more technical aspects were addressed. The ontology is modeled using OntoUML and transformed into an OWL implementation (based on the UFO implementation called gUFO), shared using FAIR principles [59]<sup>9</sup>. The ontology is divided into three components: (i) *capability core concepts sub-ontology*, (ii) *capability manifestation sub-ontology*; and, (iii) *capability type sub-ontology*.

**Capability Core Sub-Ontology.** Based on the distinctions presented in Section 2, we consider capabilities to be dispositions distinguished by the *beneficial* impact they may cause (to a subject) when they manifest. E.g., the disposition to sell is beneficial to both the supplier and the customer, the disposition to walk is beneficial to its bearer, and the disposition of a bus to carry passengers is beneficial to them. Dispositions that do not have this beneficial aspect, such as vulnerabilities, can generate undesirable impacts when manifested (e.g., the fragility of a vase).

Some dispositions may have this beneficial impact by their nature, such as the capability to walk, which is basically always beneficial to its bearer. However, others

<sup>6</sup> <http://purl.org/disc>

<sup>7</sup> The list of all competency questions are available at: <https://purl.org/creon/cq/>

<sup>8</sup> Note that, in a novel reference ontology such as *Creon*, competency questions and research questions coincide to a certain extent. Ultimately, these questions are addressed by the ontology artifact, as it embodies the knowledge uncovered through research.

<sup>9</sup> <http://purl.org/creon>

may have beneficial or negative impacts (even at the same time) depending on the context (or perspective) and subject. E.g., the cutting disposition of a knife may have positive or negative impacts, since it may be useful for cooking but may injure a person. Thus, we consider the capability type to have rigid and anti-rigid aspects [21], i.e., there are dispositions that are necessarily capabilities, and other dispositions that are capabilities only in certain contexts and with respect to certain agent’s intentions (goals).

Figure 1 presents a fragment of the proposed ontology. It shows “Capability” as a subtype of “Disposition”. “Capability” is considered a *«mixin»*, as it applies necessarily to some of its instances and contingently to others.

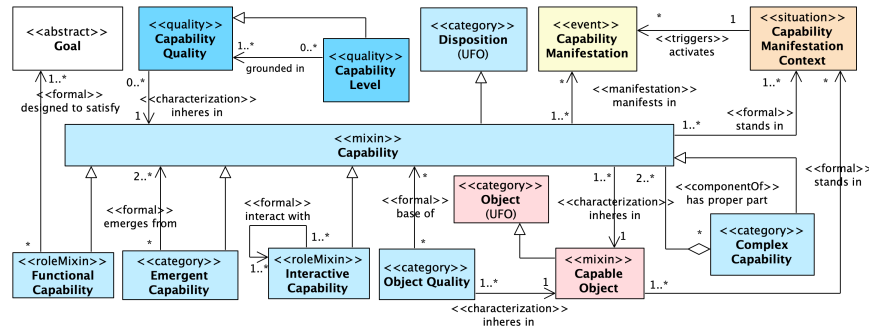


Fig. 1. Capability Ontology - Core Concepts

Capabilities inhere in *capable objects*. Capable objects include agents, non-agentive objects, functional complexes (as systems of any kind), and also “simple” (or atomic) objects. Examples include people, animals, teams, organizations, software, and other engineered systems such as electronic and mechanical devices. Objects may be capable at a given time but may cease to be so at another time. Such distinctions are represented in Figure 1. As illustrated, “Capable Object” is a subtype of “Object” (from UFO). Capable objects are characterized by one or more capabilities; that is, an object must have at least one capability to be considered an instance of “Capable Object”. “Capable Object” is represented as a *«mixin»*, once more, given that there objects that bear capabilities essentially and others only contingently.

By “benefiting” a capability subject, we mean that when a capability  $c_1$  manifests through the event  $m_1$ , it generates an outcome  $o_1$  that positively impacts the capability subject  $s_1$  (according to its intention  $i_1$ ). Dispositions that are designed with an explicit connection to a *goal* (i.e., to the propositional content of agents’ intentions) are termed *Functional Capabilities*.

The benefit brought by capabilities is always associated with their manifestation. In this context, the degree or level at which a capability benefits the capability subject may vary depending on different aspects that characterize such capability and its level (graduation or degree of development). E.g., a wooden stool has the capability to support a user sitting on it. This capability is based on the strength of the wood and the shape of the stool. The stool’s capability can be fragile or very resistant, depending on the stool’s

strength. Or, another example, the capability of software developer (or even a company) can be at the beginner or advanced level. The capability level can evolve with time. This means that the value assigned to the capability level of an individual can change (can increase or decrease). E.g., a knife may become dull due to the time it has been used but it can become sharper if sharpened. Hence, as shown in Figure 1 capabilities are characterized by a *capability level*, which corresponds to the degree to which such a capability benefits the capability subject. Based on UFO, “Capability Level” is a «*quality*» kind, and is associated with a *quality value*, which can vary over time. Capabilities can also have other qualities. E.g., the capability of an organization to produce a product can be measured by the quality “products produced per day”; the stool’s capability can have different qualities, such as the “maximum weight” it can support. This aspect is represented in Figure 1 as a *capability quality*, which characterizes capabilities. Capability qualities can also “ground” a specific capability level as, e.g., the maximum output of translating 20 texts a day associated to a Dutch translation capability corresponds to an intermediate level of that capability, while having a maximum output of 100 texts a day corresponds to have that same capability at an advanced level. A capability quality  $q_1$  grounds a capability level  $l_1$  only if  $q_1$  and  $l_1$  are inherent to the same capability  $c$ .

Being dispositions, capabilities can also be composed of other capabilities (including other dispositions); *emergent capabilities* can also “emerge from” other capabilities (including other dispositions); and, finally, *interactive capabilities* can also “interact with” other capabilities, including other dispositions. Regarding the interaction between capabilities, we use the term “interacts with” as an abstract supertype to these interactive relations between capabilities. These relations (omitted in the figure) include relations of reciprocity, complementarity, enabling and disabling, changing, among others, in line with the disposition ontology, *Disc-O*<sup>10</sup>.

Capabilities (like other dispositions) are typically grounded in some qualities called the “base” of those dispositions. E.g., the water solubility of sugar is grounded on some qualities of the sugar (its crystalline structure), or the electrical conductivity of a wire is grounded on some qualities of material constituting the wire (e.g., a copper wire is constituted by copper which has a large quantity of dissociable electrons in its outer electron shell). In Figure 1, this is represented by a «*formal*» relation between a capability inhering in a “Capable Object” and a quality inhering in that very same object that serves as a base for that capability. Thus, object quality  $q$  is only the basis of capability  $c$  if  $q$  and  $c$  characterize the same object  $o$ .

**Capability Manifestation Sub-Ontology.** Since capabilities are *dispositions*, a capability may be manifested in *capability manifestations*. These are events in which the capability is manifested. As a disposition, a capability is activated by a *capability manifestation context*. This context corresponds to the physical and temporal context, including all conditions and needed objects (i.e., resources), called here *enabler objects*. So, for this reason, a capability manifestation context is a «*situation*».

These enabler objects are objects that play the role of supporting the capable object. The enabler object (which may also be a capable object) has dispositions that interact with the capabilities of the capable object, enabling it to manifest its capabilities. Ex-

<sup>10</sup> <http://purl.org/disc>

amples of enabler objects supporting capable objects include a paintbrush supporting a painter, a car supporting a driver, or hardware infrastructure supporting a software development company. In all these examples, the dispositions of the enabler objects are essential for the manifestation of the capable objects' capabilities. In these cases, the dispositions of enabler objects and capable object interact in order to manifest together. Objects can act as an enabler object at one point in time and then stop being one at another. Moreover, objects are enablers to other objects. Enabler object has a role-like nature being an anti-rigid (contingent) and relational type. So, "Enabler Object" is represented as a *roleMixIn* of objects in Figure 2.

In addition to *enabler objects*, there are also objects that can serve as inputs in the manifestation of a capability, represented as *capability manifestation input*. Unlike enabler objects, these inputs are not connected to the capable object, and do not play a specific role regarding it (i.e., as a component of a system). Instead, they just play a *historical role* in participating in the "capability manifestation" event, being in many cases destroyed in it. Inputs are "raw materials" necessary for the capability manifestation, and include items that "flow" as information carriers, matter, or energy. Examples are the data files necessary for a processing software to work, the energy needed for a blender to manifest its capability of mixing, and the water and coffee powder needed for the coffee machine to manifest its capability to produce coffee. Figure 2 depicts these ontological distinctions.

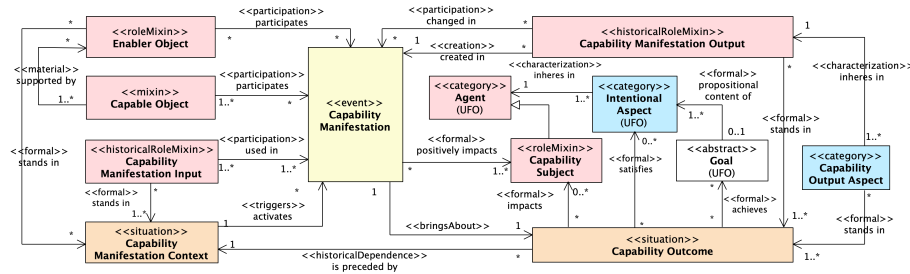


Fig. 2. Capability Ontology - Capability Manifestation

Due to the teleological nature of capabilities (and their potential benefits to a subject), one of the most important aspects related to capabilities is their outcomes. Outcomes include different types of results, outputs, effects, and impacts generated by the manifestation of the capability. This aspect is not only related to a specific output (i.e., a product) but also to the entire situation generated, which includes qualitative aspects (related to the perception of the subject), temporal aspects (when the result was generated), how it was generated, and also aspects related to the outputs (qualities and dispositions). We summarize all these results to capability subject (beneficial or not) through the concept of *capability outcome*. Based on UFO, a *capability outcome* is also considered a *situation* that brings benefits to the capability subject, as depicted in Fig. 2.

The manifestation of a capability can also produce (or change) a *capability manifestation output*. Some of these outputs represent "desired" results for the capability subject

(not all of them, e.g., residues, waste, and by-products). In this context, the output represents a “historical role” related to the participation (creation or change) of an object in the capability manifestation event. Furthermore, the output of a manifestation can have different aspects (represented as *capability output aspect*), including dispositions and capabilities, which satisfy the capability subject. E.g., the software development capability of a team or company is primarily characterized by generating software (output) that is functional, usable and safe (output aspects) for a client (capability subject).

The capability subject is an agent that can be, e.g., just a person or an entire organization. As an agent, the capability subject performs actions and can even be a capable object. As an agent, it has intentional aspects such as intentions, beliefs and desires. As previously explained, the capability subject may be impacted (mainly positively) by the manifestation of a capability. This impact occurs based on the intentions of such capability subject but also due to qualitative changes the subject may suffer. E.g., a medication may have a positive impact on a patient by reducing a headache, but it may also have negative impacts (side effects) on the patient as well (even at same time); or it can also have no effect at all. Capability subjects, because they are agents, can even share the same goal (propositional content of one or more intentions), that can also be satisfied by the capability manifestation. Figure 2 represents what was discussed above. As depicted, a “Capability Subject” is a «roleMixin» played by an *agent* (from UFO). As agents, capability subjects have *intentional aspects* (from UFO), including *intentions* (omitted in the figure) whose contents are *goals* (from UFO). When a capability is manifested, it generates a capability outcome (possibly with capability outputs), which positively impacts one or more capability subjects. Such an impact is represented by the «formal» relationship in which a capability outcome “positively impacts” the capability subject. Such a positive impact occurs together with the satisfaction of (one or more) intentions of the capability subjects, represented by the («formal») “satisfies” relationship. As a consequence, a capability outcome can also “achieve” (none or more) goals. As depicted, in addition to impacting through an outcome, the manifestation of a capability can also positively impact the capability subject directly. This is represented by the («formal») “positively impacts” relationship.

The model of Figure 2 is subject to a number of important constraints: a capability manifestation  $m$  of a capability  $c$  is only activated by a capability manifestation context  $cx$  if  $c$  stands in  $cx$ ; an enabler object  $r$  only participates in a capability manifestation  $m$  along with a capable object  $o$  if the enabler object  $r$  supports  $o$ ; a capability manifestation input  $ci$  only participates in a capability manifestation  $m$  activated by a capability manifestation context  $co$  if the input  $ci$  stands in  $co$ ; an enabler object  $or$  only participates in a capability manifestation  $m$  activated by a capability manifestation context  $co$  if the resource  $r$  stands in  $co$ ; a capability manifestation output  $o$ , created or altered by the capability manifestation  $m$ , necessarily participates in the capability outcome  $oc$  generated by the manifestation  $m$ .

**Capability Type Sub-Ontology.** When we say that a capability may cause a beneficial impact, this is a potentiality, even independently of the capability’s manifestation. Thus, to determine whether a disposition  $d$  would be a capability or not, sometimes it is also considered the type level, that is, whether the disposition  $d$  instantiates some capability type  $t$ . Figure 3 illustrates the sub-ontology of capability type. As depicted,



another time. Additionally, capability types have comparative relationships based on their level of complexity, represented by the relationship “greater complexity”.

Capability Phase not only represents stages of evolution of a capability, but also other modes of functioning and manifestation of it. E.g., due to technical problems, an automatic car may have problems avoiding obstacles in some situations. Thus, e.g., such a capability may instantiate a “malfunctioning” capability phase or even an “inactive” one. Capability Phases also classify the states of a capability. E.g., the moving capability of a car can be “not manifesting” or “manifesting”, if the car is stationary or moving.

**Ontology Evaluation** For the verification of the proposed ontology, each of the competence questions is mapped to the corresponding concepts and to the relationships between such concepts of the ontology, generating a traceability table<sup>11</sup>, as suggested in [2]. *Creon* is validated through instantiation in distinct scenarios and comparison with the capability modeling metamodel’s constructs, especially in the EA context. Main sources from the capability literature for *Creon* include [3, 17, 26, 33, 34, 37, 39, 48, 52, 53, 56, 57]. As suggested in [2], we create ontology instantiation tables related to real situations (with distinct types of capable objects, such as simple objects, agents, complex systems, etc.)<sup>12</sup> In order to demonstrate the applicability of the ontology for conceptual modeling, we use it in a concrete modeling task, using the Uber scenario, as presented in the sequel.

Uber is a digital ride-hailing platform that forms an ecosystem composed of numerous passengers and drivers, the latter operating with their own vehicles. The interactions among these human agents, physical artifacts, and the Uber software system gives rise to system-level dispositions such as ride-hailing capability. Figure 4 illustrates this case using OntoUML, showing how the *passenger transportation capability* emerges from the combination of other capabilities. As depicted, *driving capability* (of the driver) and the car’s *being driven capability* contribute to the emergence of the *passenger transportation capability*. In this case, *driving capability* and *being driven capability* are reciprocal, since they need each other to manifest.

The *passenger transport capability* is enabled by the *ride coordination capability* of the *ride coordinator system* (performed by the Uber software system). In this case, the Uber software system enables the meeting between the passenger and the driver, providing the necessary information, as well as helping with navigation. The emergent capability of *passenger transportation* is reciprocal to the *passenger’s being transported susceptibility*. These reciprocal relationships reflect the mutual dependencies between these dispositions. The figure also highlights certain vulnerabilities: the *car’s susceptibility to breaking* can disable its “*being driven capability*” and prevent the transportation of the passenger. The *Uber software system’s susceptibility to fail* (due to technical issues) may disable its *capability to coordinate rides*.

In this context, as depicted, the passenger is the *capability subject*, and the driver in this context is a *capable object*, characterized by the *capability to drive*. As illustrated in Figure 4, the *passenger transportation capability* is manifested through the event “*transport passenger*”, which constitutes a *capability manifestation*. This event is triggered by a situation—a *capability context*—in which the passenger is located at an origin point

<sup>11</sup> <https://purl.org/creon/cq/>

<sup>12</sup> URL of the instantiation table: <https://purl.org/creon/validation>

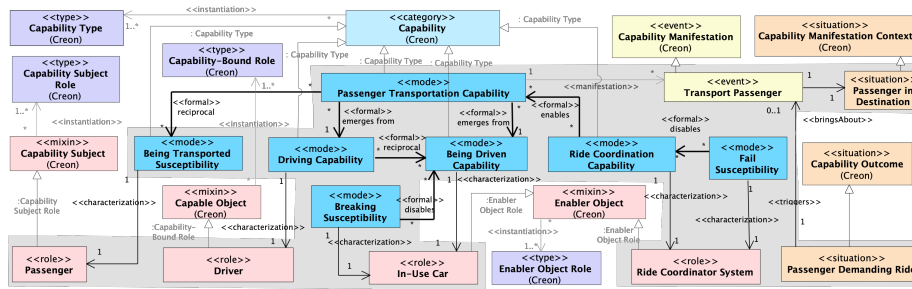


Fig. 4. Ontology Validation: Uber's Scenario

and wishes to reach a destination (named *passenger demanding ride* situation). In order to transport the passenger to the desired location, the driver must know the originating point, who the passenger is, and the intended address. This information, in this case, is considered a *capability manifestation input* (not depicted).

As a result of the action of transporting the passenger to their destination, a new situation is brought about: the “*passenger being at the destination location*”. This situation, as it results from a capability manifestation, is classified as a *capability outcome*, since it satisfies the intentions of the passenger (capability subject). An important aspect of this example is that both the *car* and the *Uber software system* work as *enabler objects* in this scenario (in addition to also being *capable objects* as well), as they support the driver (the *capable object*) in the passenger transport process. Notably, the distinction between *capable object* and *enabler object* is dependent on the perspective adopted in the analysis of a given scenario.

#### 4 Implications for the Analysis and Design of System Capabilities

The proposed distinctions can support a better understanding of how capabilities of distinct nature (from people, hardware, software, etc) interact and contribute to the emergence of IS and organizational capabilities in distinct platform implementations, such as CPS and IoT. One implication of the ontology includes the redesign of modeling notations. Concerning this, the ontology can be used to improve, e.g., requirement models in software and system engineering areas. Requirements generally represent the desired (functional and non-functional) capabilities, i.e., those capabilities are required for achieving stakeholders' goals (and must interact for this). Like capabilities, requirements are often structured hierarchically, decomposed into different levels, and related to allow traceability. Considering that a system's requirements generally refer to capabilities, the relationships between the capabilities proposed in this study could be applied to improve the modeling of required capabilities, improving traceability, design, and implementation of the capabilities.

The same idea is also applicable to use case diagrams. They also essentially represent the functional capabilities of systems. *Creon*, in this case, can also improve how to decompose use cases and how to relate them. In fact, as dispositions (capabilities)

integrate structural and behavioral aspects, the proposed ontology can contribute to supporting the modeling of behavioral aspects as well. Widely used notations such as UML and SysML incorporate the concept of “behavioral features” such as *operations* and *methods*. Methods and operations are similar to the concept of disposition (and also capability), as they represent characteristics that are manifested through actions, using inputs, and generating outputs. However, modeling operations in such notations is often simplistic and does not consider how these operations can interact, be combined, or form other operations. *Creon* can support the development of contributions along this line. It can underpin relationships between operations, especially in the context of emergence, improve the modeling of interactive diagrams (e.g., sequence diagrams), and improve the understanding of the behavior resulting from these operations’ interactions.

*Creon* can also be used to define the temporal order of activities and tasks in behavioral diagrams, such as activity diagrams and sequence diagrams (UML/SysML). The flow of activities is essentially a manifestation of the capabilities of these actors. Consequently, the way the activities flow unfolds depends on the relationships between the dispositions and capabilities of the actors involved. E.g., if one capability enables another, this will be reflected as sequential actions in the activity flow. By representing dispositions and capabilities and their relations, it would be possible to infer, validate, verify, or even support the construction of these behavioral diagrams.

The distinctions proposed in the capabilities ontology can also impact the structural modeling of Information Systems (IS). In this context, a connection between entities or components of the IS can only exist if they have interactive capabilities that allow their interaction, that is, if their capabilities are related. In this way, capability models based on the proposed distinctions can help in the elaboration, validation, and verification of structural models of different types in the IS context.

## 5 Related Work

Related works that use foundational ontologies in the context of EA modeling include [6,43,50]. Both use UFO to perform ontological analysis of ArchiMate’s constructs. For instance, the work approached by Azevedo et al. [6] conducts an ontological analysis of capability, but it concentrates on strategic elements. In this ontological analysis, which we adopt and expand upon in the current work, the authors briefly address the definition of capability. The author suggests using ArchiMate to enhance Enterprise Modeling as an application, even permitting the language to represent capabilities. Nevertheless, no capability ontology was put forth in that work. In addition, the ontological analysis did not take capability relationships into account.

The work presented in [28] presents a capability ontology within the context of the manufacturing industry, seeking to provide a conceptual framework for understanding and representing organizational capabilities. Furthermore, it emphasizes that capabilities are related not only to people’s individual skills but also to resources of different types. However, capabilities there are not considered as possibly complex entities. Although they have specific restrictions and properties, the phenomena of emergence or interaction between capabilities within that ontology are not addressed. Similarly, the work proposed in [31] also contributes in this sense by proposing a business capabil-

ities ontology. In that ontology, capabilities are subdivided into internal and external, reflecting their relationship with business processes and services. These capabilities are defined as a combination of abilities and capacities. The work [31] emphasizes that capabilities are complex entities, susceptible to being composed of others. A fundamental aspect therein is the way in which capabilities are related and interconnected. They are connected through collaborative relationships. Despite considering elements such as context, outputs, relationships between capabilities, and their composition, the phenomenon of emergence is not considered. The interactions between the capabilities are also not refined into sub-relationships as we do here.

The work of [30] contributes to the understanding of organizational capabilities by presenting a conceptual model that connects capabilities to the organization’s objectives and metrics. Among the related works, this one stands out for its consideration of the types of capabilities, including some important classifications, such as the distinction between primary capability and enabling capability. Furthermore, the work introduces the idea of a relationship of influence between capabilities, recognizing that these capabilities do not exist in isolation, but dynamically interact with each other in the organization. However, like previous works, this research does not delve into the different types of relationships between capabilities, nor does it consider the phenomenon of emergence, which plays a crucial role in understanding organizational dynamics.

The proposal shown in [15] also offers a contribution in this area by proposing an ontology that addresses the concepts of function, behavior, and capability. It provides an understanding of the capabilities inherent to artifacts and their relationship with the functions performed by these artifacts. Therefore, one of its main contributions is representing the interconnection between the capabilities and functions of artifacts. Although the work offers a well-founded understanding of capabilities, it does not explore how these capabilities interact and influence each other. Furthermore, the emergence and composition of capabilities, that is, how capabilities can be combined to form more complex capabilities, are also not comprehensively addressed there.

**Table 1.** Related Works Comparison

<b>RELATED WORKS</b>	<b>Interaction</b>	<b>Composition</b>	<b>Cap. Type</b>	<b>Obj. Quality</b>	<b>Manifestation</b>
<i>Köcher et al. [28]</i>				X	X
<i>Loucopoulos et al. [31]</i>	O	X			
<i>Kudryavtsev et al. [30]</i>	O		X		
<i>Järvenpää et al. [25]</i>		X			X
<i>Koutsopoulos et al. [29]</i>	X				X
<i>Compagno et al. [15]</i>				X	O

The ontology proposed in [25] proposes a capability ontology in the context of manufacturing resources. An important feature of that ontology is allowing capabilities to be combined with each other. Furthermore, capabilities are related to a specific context (workspace). Another relevant aspect is the relationship between capabilities, which is established through inputs and outputs. However, despite the introduction of the concept of combined capability, the phenomenon of emergence was also not considered

there. Moreover, the relationships between capabilities are refined into more specific relationships as we do here. Table 1 summarizes the comparison with the related works mentioned above. Each column represents an aspect compared above that the related work satisfies totally (X) or partially (O). As shown, no related work addresses the emergence phenomenon but just the composition.

## 6 Final Remarks

In this paper, we propose *Creon*, a reference ontology for capabilities. Key concepts associated with capability, including capability manifestation, output, outcome, level, capability types, and so forth, were defined by the ontology. To explain their emergence and interaction, *Creon* drew on system science and disposition theory. Based on UFO distinctions, this ontology was proposed using the OntoUML notation. An illustration in the context of transportation was used to demonstrate the application of the ontology. We discussed capability interaction and the emergence, and demonstrated how these aspects contribute to the manifestation of behaviors. Furthermore, we conducted an analysis of the different concepts of capability found in the literature, specifically those pertaining to tasks in capability modeling in the context of ISs, in order to establish the requirements for our methodology.

Concerning future works, the evolution of capability over time is a topic that we intend to further elaborate on. This is possible in these foundations given that capabilities are endurants and can change qualitatively while maintaining their identity. This was explored partially with the capability level concept, but it is important to further explain how complex capabilities alter their composition and relationships. Using the system ontology to support system capability representation for particular system types related to IS context would be a significant future endeavor. For these kinds of systems in the IS context, the relationships between dispositions (and also capabilities) that are discussed in this work can be made more specific. In complex contexts involving various types of systems, such as system-of-systems, cyber-physical systems, or digital twins, an extension of ontology can be useful for describing system capabilities and digital requirements. Finally, we plan to look into how ontology can help with pattern recognition methods that find capability emergence patterns in ISs and organizational data, particularly in intricate network models.

**Acknowledgments.** This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001, CNPq (443130/2023-0, 313412/2023-5), FAPES (129/2021-P:2021-GL60J, 368/2022-P:2022-NGKM5, 1022/2022, 993/2023 P: 2023-N3HJD), and the DSYNE INTPART network (Research Council of Norway project number 309404).

## References

1. Ackoff, R.L.: Towards a system of systems concepts. *Management Science* **17**(11), 661–671 (Jul 1971)

2. de Almeida Falbo, R.: Sabio: Systematic approach for building ontologies. In: Guizzardi, G., Pastor, O., Wand, Y., de Cesare, S., Gailly, F., Lycett, M., Partridge, C. (eds.) Proceedings of the 1st Joint Workshop ONTO.COM / ODISE on Ontologies in Conceptual Modeling and Information Systems Engineering co-located with 8th International Conference on Formal Ontology in Information Systems, ONTO.COM/ODISE@FOIS 2014, Rio de Janeiro, Brazil, September 21, 2014. CEUR Workshop Proceedings, vol. 1301. CEUR-WS.org (2014), [http://ceur-ws.org/Vol-1301/ontocomodise2014\\_2.pdf](http://ceur-ws.org/Vol-1301/ontocomodise2014_2.pdf)
3. Antunes, G., Borbinha, J.: Capabilities in systems engineering: an overview. In: Exploring Services Science: 4th International Conference, IESS 2013, Porto, Portugal, February 7-8, 2013. Proceedings 4. pp. 29–42. Springer (2013)
4. Aquino, N.M.R.: A smart assessment of business processes for enterprises decision support. Ph.D. thesis, Université de Lorraine (2021)
5. Azevedo, C.L.B., et al.: An Ontology-Based Well-Founded Proposal for Modeling Resources and Capabilities in ArchiMate. In: 17th IEEE International EDOC Conference (EDOC 2013). pp. 39–48. IEEE Computer Society Press (2013)
6. Azevedo, C.L.B., Iacob, M., Almeida, J.P.A., van Sinderen, M., Pires, L.F., Guizzardi, G.: Modeling resources and capabilities in enterprise architecture: A well-founded ontology-based proposal for ArchiMate. *Information Systems* **54**, 235–262 (2015). <https://doi.org/10.1016/j.is.2015.04.008>
7. Barton, A., et al.: A taxonomy of disposition-parthood. In: Workshop on Foundational Ontology in Joint Ontology Workshops: JOWO 2017. vol. 2050, pp. 1–10. CEUR-WS: Workshop proceedings (2017)
8. von Bertalanffy, L.: *General system theory*. George Braziller, New York (1968)
9. Bunge, M.: *Treatise on Basic Philosophy: The Furniture of the World. Ontology I*. Reidel Pub. (1977)
10. Bunge, M.: *Treatise on Basic Philosophy. Ontology II: A World of Systems*. Springer Netherlands, Dordrecht, Netherlands (1979)
11. Bunge, M.: *Emergence and convergence: Qualitative novelty and the unity of knowledge*. University of Toronto Press (2003)
12. Calhau, R.F., Almeida, J.P.A., Kokkula, S., Guizzardi, G.: Modeling competences in enterprise architecture: from knowledge, skills, and attitudes to organizational capabilities. *Software and Systems Modeling* pp. 1–40 (2024)
13. Calhau, R.F., Prince Sales, T., Oliveira, Í., Kokkula, S., Ferreira Pires, L., Cameron, D., Guizzardi, G., Almeida, J.P.A.: A system core ontology for capability emergence modeling. In: International Conference on Enterprise Design, Operations, and Computing. pp. 3–20. Springer (2023)
14. Calhau, R.F., et al.: Towards Ontology-based Competence Modeling in Enterprise Architecture. In: 2021 IEEE 25th International Enterprise Distributed Object Computing Conference (EDOC). IEEE (Oct 2021)
15. Compagno, F., et al.: Towards a formal ontology of engineering functions, behaviours, and capabilities. *Semantic Web Journal* pp. 285–318 (2024)
16. Dogan, M.H., Henshaw, M., Johnson, J., Harding, A., CWG, I.U.: *Capability engineering ontology*. INCOSE UK report (2012)
17. Dori, D., Sillitto, H.: What is a system? an ontological framework. *Systems Engineering* **20**(3), 207–219 (May 2017). <https://doi.org/10.1002/sys.21383>
18. Feng, Y., Zou, Q., Zhou, C., Liu, Y., Peng, Q.: Ontology-based architecture process of system-of-systems: From capability development to operational modeling. *Applied Sciences* **13**(9), 5419 (2023)
19. Galton, A., et al.: Dispositions and the infectious disease ontology. In: *Formal Ontology in Information Systems: Proceedings of the Sixth International Conference (FOIS 2010)*. vol. 209, p. 400. Ios Press (2010)

20. Guizzardi, G., Wagner, G., Almeida, J.P.A., Guizzardi, R.S.S.: Towards ontological foundations for conceptual modeling: The unified foundational ontology (UFO) story. *Applied Ontology (Online)* **10**, 259–271 (2015). <https://doi.org/10.3233/A0-150157>
21. Guizzardi, G.: Ontological Foundations for Structural Conceptual Models. No. 15 in *Telematica Institute Fundamental Research Series*, Telematica Instituut, Enschede, The Netherlands (2005)
22. Guizzardi, G., et al.: Towards ontological foundations for the conceptual modeling of events. In: *Conceptual Modeling*, pp. 327–341. Springer Berlin Heidelberg (2013)
23. Guizzardi, R.S., Li, F.L., Borgida, A., Guizzardi, G., Horkoff, J., Mylopoulos, J.: An ontological interpretation of non-functional requirements. In: *FOIS*. vol. 14, pp. 344–357 (2014)
24. Henshaw, M.J.d.C., Lister, P., Harding, A.D., Kemp, D., Daw, A.J., Farncombe, A., Touchin, M.: 6.3. 2 capability engineering—an analysis of perspectives. In: *INCOSE International Symposium*. vol. 21, pp. 712–727. Wiley Online Library (2011)
25. Järvenpää, E., Siltala, N., Hylli, O., Lanz, M.: The development of an ontology for describing the capabilities of manufacturing resources. *Journal of Intelligent Manufacturing* **30**(2), 959–978 (2019)
26. Josey, A.: *TOGAF® version 9.1-A pocket guide*. Van Haren (2016)
27. Juarrero, A.: Dynamics in action: Intentional behavior as a complex system. *Emergence* **2**(2), 24–57 (2000)
28. Köcher, A., Belyaev, A., Hermann, J., Bock, J., Meixner, K., Volkmann, M., Winter, M., Zimmermann, P., Grimm, S., Diedrich, C.: A reference model for common understanding of capabilities and skills in manufacturing. *at-Automatisierungstechnik* **71**(2), 94–104 (2023)
29. Koutsopoulos, G., Andersson, A., Stirna, J., Henkel, M.: Application and evaluation of interlinked approaches for modeling changing capabilities. *Software and Systems Modeling* pp. 1–30 (2024)
30. Kudryavtsev, D., Grigoriev, L., Bobrikov, S.: Strategy-focused and value-oriented capabilities: Methodology for linking capabilities with goals and measures. In: *CEUR Workshop Proceedings*. vol. 1182, pp. 15–26 (2014)
31. Loucopoulos, P., Kavakli, E.: Capability modeling with application on large-scale sports events. In: *AMCIS*. vol. 2016, pp. 11–13 (2016)
32. Maier, J.: Abilities. In: Zalta, E.N., Nodelman, U. (eds.) *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Fall 2022 edn. (2022)
33. Martin, J.: *Enterprise capability management* (2021)
34. Martin, J., Faisandier, A.: *Enterprise systems engineering background* (2021)
35. Martin, J.N., O’Neil, D.P.: Enterprise architecture guide for the unified architecture framework (uaf). In: *INCOSE International Symposium*. vol. 31, pp. 242–263. Wiley Online Library (2021)
36. Martin, J., Axelsson, J., Carlson, J., Suryavedara, J.: The capability concept in the context of systems of systems: a systematic literature review. In: *2022 IEEE International Symposium on Systems Engineering (ISSE)*. pp. 1–8. IEEE (2022)
37. Merrell, E., et al.: *Capabilities* (2022)
38. Molnar, G., Bradley, N.: *Powers: A study in metaphysics*. Clarendon Press (2003)
39. Morgan, P.: The concept of capacity (draft version). *Study on capacity, change and performance* pp. 1–19 (2006)
40. Mossio, M., et al.: Emergence, closure and inter-level causation in biological systems. *Erkenntnis* **78**(S2), 153–178 (2013)
41. Mumford, S., Anjum, R.: *Getting Causes from Powers*. Oxford University Press (2011)
42. Mumford, S., Anjum, R.L.: Powers of wholes and parts. In: *FOIS*. p. 4 (2016)
43. Nardi, J.C., de Almeida Falbo, R., Almeida, J.P.A., Guizzardi, G., Pires, L.F., van Sinderen, M.J., Guarino, N., Fonseca, C.M.: A commitment-based reference ontology for services.

- Information Systems **54**, 263–288 (Dec 2015). <https://doi.org/10.1016/j.is.2015.01.012>
44. Naudet, Y., et al.: Towards a systemic formalisation of interoperability. *Computers in Industry* **61**(2), 176–185 (feb 2010)
  45. O’Connor, T.: Emergent properties. *American Philosophical Quarterly* **31**(2), 91–104 (1994)
  46. Oliveira, Í., Sales, T.P., Baratella, R., Fumagalli, M., Guizzardi, G.: An ontology of security from a risk treatment perspective. In: International conference on conceptual modeling. pp. 365–379. Springer (2022)
  47. OMG: Unified Modeling Language (OMG UML) version 2.5.1. techreport, Object Management Group (OMG) (Dec 2017), <http://www.omg.org/spec/UML/2.5.1>
  48. Pyster, A., Olwell, D.H., Hutchison, N., Enck, S., Anthony Jr, J.F., Henry, D., et al.: Guide to the systems engineering body of knowledge (sebok) v. 1.0. 1. Guide to the Systems Engineering Body of Knowledge (SEBoK) (2012)
  49. Sales, T.P., Almeida, J.P.A., Santini, S., Baião, F., Guizzardi, G.: Ontological analysis and redesign of risk modeling in ArchiMate. In: 22nd IEEE Intl EDOC Conf (EDOC 2018). pp. 154–163 (2018)
  50. Sales, T.P., Baião, F., Guizzardi, G., Almeida, J.P.A., Guarino, N., Mylopoulos, J.: The common ontology of value and risk. In: Conceptual Modeling: 37th International Conference, ER 2018, Xi’an, China, October 22–25, 2018, Proceedings 37. pp. 121–135. Springer (2018)
  51. Sarkar, A., Šormaz, D.: Ontology model for process level capabilities of manufacturing resources. *Procedia Manufacturing* **39**, 1889–1898 (2019)
  52. Saxena, M.: Capability Management. Global India Publications (2009)
  53. Sillitto, H.G.: 10.2.1 “Composable Capability” - Principles, strategies and methods for capability systems engineering. *INCOSE International Symposium* **23**(1), 723–738 (Jun 2013). <https://doi.org/10.1002/j.2334-5837.2013.tb03050.x>
  54. Specification, O.: System modeling language (sysml) specification (2006)
  55. Spencer-Smith, R.: Reductionism and emergent properties. In: Proceedings of the Aristotelian Society. pp. 113–129. JSTOR (1995)
  56. Tell, A.W.: What capability is not. In: Perspectives in Business Informatics Research: 13th International Conference, BIR 2014, Lund, Sweden, September 22-24, 2014. Proceedings 13. pp. 128–142. Springer (2014)
  57. The Open Group: TOGAF Business Capabilities Guide V2. <https://pubs.opengroup.org/togaf-standard/business-architecture/business-capabilities.html> (2022), [Accessed 07-12-2024]
  58. The Open Group: Archimate 3.2 specification (2023), <https://pubs.opengroup.org/architecture/archimate3-doc/>
  59. Wilkinson, M.D., Dumontier, M., Aalbersberg, I.J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.W., da Silva Santos, L.B., Bourne, P.E., et al.: The fair guiding principles for scientific data management and stewardship. *Scientific data* **3**(1), 1–9 (2016)
  60. Yilma, B.A., et al.: Systemic formalisation of cyber-physical-social system (CPSS): A systematic literature review. *Comput. Ind.* **129**, 103458 (2021)