




Ontological Analysis of Advanced Capability Modeling in ArchiMate: A First Step Towards Language Revision

Rodrigo F. Calhau^{1,2,3} , João Paulo A. Almeida¹ , and Giancarlo Guizzardi³ 

¹ Ontology & Conceptual Modeling Research Group, Fed. Univ. of Espírito Santo, Brazil

² LEDES, Federal Institute of Espírito Santo, Serra, Brazil

³ Semantics, Cybersecurity & Services, University of Twente, Enschede, The Netherlands
calhau@ifes.edu.br; jpalmeida@ieee.org; g.guizzardi@utwente.nl

Abstract. In order to support capability management, the field of Enterprise Architecture proposes methods and notations to model enterprises and their capabilities. ArchiMate is one of such notations and includes constructs to support capability mapping and other capability management tasks. However, the notation lacks some fine grained distinctions which are required to understand intricate phenomena involving capabilities, including capability *interaction* and the *emergence* of capabilities. In this work, we perform an ontological analysis of the language's support for capabilities based on a well-founded ontology of capabilities aligned with the Unified Foundational Ontology (UFO). Through this ontological analysis, we identify some issues outlining possible improvements for the notation. This is a first step towards language redesign, which may include the proposal of language patterns and/or the revision of language constructs.

Keywords: ArchiMate · ontological analysis · capabilities · emergence.

1 Introduction

Enterprises and organizations have been facing challenges with the rapid and unpredictable development of new technologies that impact their operating environments. With these fast changes comes the need for the development of organization-wide capabilities, which emerge from a well-balanced combination of organizational elements.

Given the importance of organizational capabilities, it is no surprise that they have been given attention in disciplines such as capability management and Enterprise Architecture (EA) [7]. In particular, EA aids the capability management process by offering structured visualizations that align strategic goals with IT and business processes, identifying gaps and opportunities for improvement [1]. Notations such as ArchiMate [45] have been widely adopted to support practices such as business and information technology capability planning and mapping. With this notation, it is possible to construct models to support capability management and visualize important aspects such as capability decomposition and other types of capability relationships [21, 46].

Capability modeling faces a series of demanding challenges, especially with regard to how to decompose and compose capabilities, how to trace, relate (or combine), and compare them. Although currently available notations offer resources to represent some

aspects of capabilities such as relationships of composition, association, impact, dependence, and realization, they do not fully address the capability emergence phenomenon. In other words, existing notations may be effective in representing isolated capabilities, but they do not provide a robust framework for dealing with the complexity inherent in modeling capabilities in a broader context.

Some of the limitations of existing notations can be traced to their underlying ‘world view’. Making this world view explicit and accounted for is the objective of *ontological analysis*, which has been proposed as a means to evaluate the expressiveness and domain appropriateness of conceptual modeling languages [38]. In this context, ontologies can provide us with reference theories that articulate key domain distinctions; these distinctions are then reflected in modeling constructs, patterns and guidelines.

In this paper, we employ *ontological analysis* as a principled approach to assess capability modeling in ArchiMate. Our starting point is the ontology of capabilities we have proposed earlier⁴. This ontology is based on the Unified Foundational Ontology (UFO) [18] and is represented using the modeling language OntoUML [18]. It takes into account emergence [22, 32, 35, 43] and disposition theories [6, 16, 30, 33] to provide a well-founded conceptualization for capabilities and their relations.

The analysis presented in this paper builds up on our past work which proposes the representation of capability emergence in ArchiMate [10] and also has as a background the work of Azevedo et al. [4, 5] (which did not address detailed capability relationships). The approach is based on the analysis of the key conceptual notions and their relations in order to guide adequate capability representation in EA models. Domain-adequate representations are, in turn, key to supporting the use of EA models in capability-based practices. We argue that a conceptual analysis concerning capability *relationships* is also fundamental to their adequate representation in EA models. In this case, capabilities can be related as a result of the interactive relationship between business entities and professionals. Based on this, they can even give rise to new organizational capabilities in teams, departments, or other organizational structures. We address the conceptual problem by adopting the capability relationships from the ontology, reflecting hierarchical and interactive relationships based on theories of dispositions and property emergence in the literature.

This paper is structured as follows: Section 2 presents an overview of the literature related to capabilities and the Unified Foundational Ontology and the capability ontology; Section 3 presents the ontological analysis based on the capability ontology; Section 4 presents preliminary suggestions, based on the ontological analysis, of enhancement ArchiMate’s capability metamodel and specification; Section 5 discusses related work, and Section 6 concludes with our final remarks.

2 Background

2.1 Capabilities

Capability is generally defined as the “ability to do something” [31]. As noticed in most capability definitions, the meaning of *capability* is closely connected to the meaning of

⁴ Currently under review.

ability. Ability is commonly defined as the power to act, or as a kind of dispositional property that allows one to do something (useful or not) [24]. More precisely, the ability concept is defined by [24] as the “power that relates an agent to an action”. So, ability is a potentiality (power) related to a bearer and also to an action. Given this meaning, what distinguishes it from the capability? Another of the main distinctions regards the value aspect. In this sense, ability definitions are usually more generic, and not directly related to the “desired” results, outcomes, or achievements. They are more “neutral” in the sense of ability impact, i.e., abilities can be valuable (beneficial, useful) to a stakeholder or can even include “not desirable” effects (e.g., vulnerabilities). On the other hand, one essential aspect that commonly distinguishes capabilities concerns their usefulness, related to outcomes and outputs. For example, in enterprise architecture and systems engineering areas, capabilities are defined as the “ability to do something useful” [25,26,29]; in information systems area as “ability to achieve a desired effect”; and, in the military field, it is defined as “ability to achieve a determined military objective” [2]; As illustrated, the “beneficial” aspect is present in capability definitions in distinct areas. Many of these definitions were studied in [44], which generalized the capability definition. Following [44], a capability is the potential for certain outcomes to be achieved concerning certain source entities. Moreover, capabilities link these source entities (e.g., the bearer) to a result (e.g., output, outcomes, achievements) (ibid.). In summary, capabilities are changeable [14] and composable entities [42] which represent the quality of being capable of achieving specific effects or declared objectives [40].

2.2 Modeling Capabilities in ArchiMate

ArchiMate is a widely used modeling language in EA that allows the modeling of capabilities in this context. It provides a comprehensive framework for visualizing, analyzing, and communicating architectural blueprints within organizations. The notation offers a standardized way to depict, understand, and manage the complexities of enterprise architectures. It serves as a bridge between business and IT domains, enabling stakeholders to align strategic goals with operational realities through a unified visual language [45].

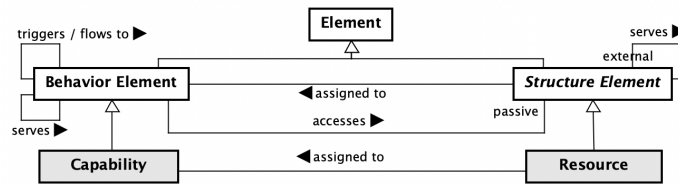


Fig. 1. Fragment of the ArchiMate’s Metamodel (Adapted from [45])

The core of ArchiMate notation is its metamodel, which categorizes architectural elements into *behavior elements* and *structure elements* [45]. This is depicted in Figure 1. In this case, behavior elements capture the dynamic aspects of an enterprise, such as

processes, events, interactions, and other behaviors. Structure elements represent the static aspects, including organizational units, roles, and equipment. ArchiMate divides structure elements into *active* structure elements and *passive* structure elements. So, it divides the elements in a similar way to natural languages, with active structural elements like subjects, behavioral elements (like verbs), and passive structural elements (like objects) [45].

ArchiMate organizes architectural models into distinct *layers*, each representing a different perspective (or viewpoint) of the enterprise architecture. These layers include the *Business Layer*, focusing on organizational services, components, functions, and processes; the *Application Layer*, addressing software applications supporting business functions and services; and the *Technology Layer*, dealing with infrastructure and hardware components. ArchiMate also considers a perspective to represent motivational elements such as goals, drivers, requirements, and so on; and also the *Strategy Layer*, focusing on business capabilities and resources [45].

Capabilities in ArchiMate metamodel are considered behavior elements (as events and processes) [45], in this case, *strategy behavior elements*. They represent an ability that a Structure Element (organization unit, person, or system) possesses. Figure 2 illustrates an example of a capability model, corresponding to the ArchiSurance case study [20]. In ArchiMate models, they provide a high-level view of the current and desired abilities of an organization. As behavior elements, they can trigger, serve (contribute to), and flow (i.e., exchange matter, energy, or information) to each other (not considered in the figure). In the strategic view, capabilities can be assigned by resources (e.g., “CRM Automation” assignment resource to “Customer Care” capability). They can also be realized by other structure or behavior elements (e.g. business actors, business roles, business processes, business function, and so on). E.g., “Customer Care” capability, in the example, is realized by the “Customer Relation” function of the “Customer Service” actor. In this case, this means that these elements can be used to achieve a specific capability. Finally, capabilities can aggregate and be composed of other capabilities. This is depicted in Figure 2 as, for example, “Marketing” capability is formed by “Marketing Development” and “Campaign Management”. In this case, composition corresponds to a whole/part relationship that expresses an existential dependency, and aggregation does not [45].

2.3 Ontological Baseline

In order to account for capability-related phenomena more precisely, we employ here a fragment of the UFO foundational ontology [17], which defines a system of domain-

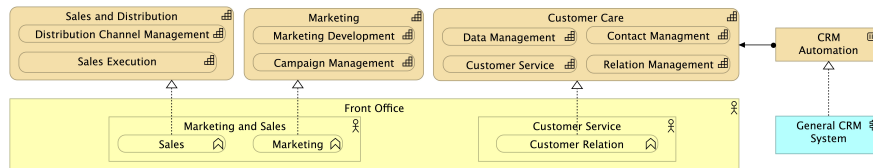


Fig. 2. Example of Capability Modeling in ArchiMate (extracted from [20])

independent categories and their ties, which can be used to articulate conceptualizations of phenomena of interest. UFO has been developed based on theories from Formal Ontology, Philosophical Logics, Philosophy of Language, Linguistics, and Cognitive Psychology [18]. The employed fragment captures the first two basic ontological categories: that of the *types* (concepts, universals, e.g., Person, Planet, Music Band) and that of the *individuals* (particular things, e.g., John, Mary, Saturn, The Beatles). Individuals include *concrete individuals* and *abstract individuals*. *Concrete Individuals* are partitioned into *events* (i.e. *perdurants*), *endurants*, and *situations*. *Events* are *individuals* that occur in time, including processes activities, actions, and tasks. Events are causally related and can be atomic or complex. Besides this, events can change, create, or terminate objects, including their aspects [19]. *Endurants* are individuals that persist in time changing qualitatively while retaining their identity (i.e., people, organizations, cars). *Endurants* include *objects* and *aspects*. An *Object* is an *endurant* that is considered existentially independent (like John, his car). Objects formed by others (performing distinct functions) are called *functional complexes*. An *Aspect* is a reified property that *inheres* in another *endurant* (termed its bearer), on which it is existentially dependent. Aspects (as full-fledge *endurants*) have a lifecycle of their own and can be created, destroyed, or otherwise changed qualitatively in time.

Of special interest to us in this work is the UFO notion of *dispositions*. *Dispositions* are *aspects* that can be manifested through the occurrence of *events* (possibly agents' actions, such as Anna's speaking English). In *situations* where *dispositions may manifest*, they are said to be "activated" (e.g., when a magnet is close to some ferrous material, or when Anna is prompted to introduce the topic of a meeting). As *endurants*, they can themselves bear aspects, and change qualitatively through time [17].

In order to incorporate the UFO distinctions, we are using OntoUML [18] as a notation for modeling the reference ontologies. OntoUML allows the creation of well-founded ontological models. It is a UML profile that incorporates the foundational distinctions proposed in UFO through stereotypes.

UFO is the foundation of many Core ontologies, modeled using OntoUML. The ontology used in this ontological analysis, based on our previous works [10, 11], was proposed as a UFO-based reference ontology about capability. Figure 3 depicts the fragment of this ontology. Based on UFO distinctions, capabilities are a «*mixin*» of "valuable" dispositions since aggregate rigid and anti-rigid aspects. Capabilities also have a *capability level* as a «quality», corresponding to their maturity. They inhere in a (non-agentive or agentive) *capable object* (including systems). *Capabilities* are manifested through a *capability manifestation*, which is triggered by *capability context* (situation) and brings about a *capability outcome* (situation). As depicted, the manifestation of a capability can also employ ('consume') a *capability input*, a «*rolemixin*» (i.e., *mixin* of roles) performed by an object. It can also produce (or change) a *capability output*. This output is a «*rolemixin*» played by an object and corresponds to some desired result for the value subject. Based on the categorical base of the disposition [12], we consider that *object qualities* contribute to capability formation (e.g., the quality of wheel shape contributes to wheel rolling capability).

Capabilities can be *interactive capabilities*, which correspond to a 'role' performed by capabilities when bearing certain relations to each other (and other dispositions).

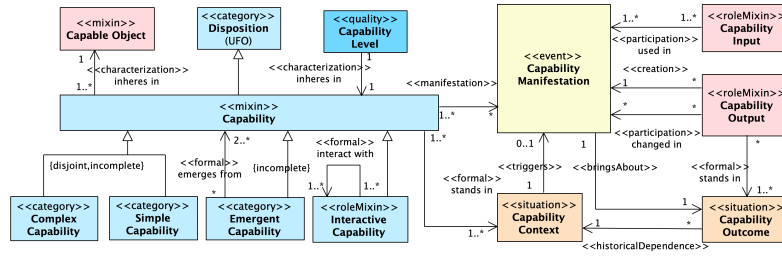


Fig. 3. Fragment of the Disposition and Capability Ontology used in the ontological analysis

These relations include relations of reciprocity [16,33], enabling [6], and changing [33], among others (not all depicted in the figure). We use the term “interacts with” as an abstract supertype to these relations. *Capabilities* can also be complex or atomic. *Complex capabilities* are those that have other *capabilities* (and dispositions) as proper parts, and *atomic capabilities* do not. *Complex capabilities* are manifested through *complex events* (i.e. processes) and *atomic capabilities* are manifested through *atomic events*. *Capabilities* inhere in *objects* (capable objects), which can be atomic objects or functional complexes, especially systems. *Capabilities* of a system include two types: *emergent capability* or *resultant capability* [11]. *Resultant capabilities* are those that come “directly” from some *particular dispositions* of components (i.e., component capabilities). For example, the breaking capability of a car comes directly from the breaking capability of the car’s break system. They can even be present in the system parts in isolation. In contrast, *emergent dispositions* are those that, while related to the (interactive) disposition of parts, are not present in isolation in the separated parts [9].

3 Ontological Analysis of Capability-Related Elements in ArchiMate

In this research, we conduct an ontological analysis of the capability element in ArchiMate by using the capability ontology presented above. Our approach involves identifying and examining the fundamental capability-related elements and their relationships within ArchiMate’s metamodel. We delve into the descriptions and meanings of these elements as specified in ArchiMate specification, aiming to understand their relevance in enterprise architecture. Then, we compare ArchiMate’s capability-related elements with capability ontology concepts, particularly focusing on UFO distinctions. This comparison aims to establish correspondences and clarify how ArchiMate’s capability modeling aligns with broader ontological frameworks. After this comparison, we identify issues regarding the capability modeling steps in ArchiMate. Next, we present the issues related to ontological analysis. For each issue, we provide context, explain how ArchiMate deals with it, and then analyze the ArchiMate approach using the capability ontology.

3.1 The Capability as an Aspect

As presented before, capability is defined in distinct fields as an *ability* of certain entity (a system, organization, enterprise, person, etc.) to do something and generate some outcome. As stated, this ability corresponds to a “power”, “propensity”, or “potentiality” of the bearer, i.e., a potential to act or behave. Then, when an activating situation happens, the capability manifests itself as an event or behavior. The literature, mainly related to disposition theories, makes a clear distinction between the capability (as a potentiality) and its manifestation (the event).

ArchiMate issues As presented, in ArchiMate, capabilities are categorized as behavior elements, as addressed, similar to events and processes, related to the organization’s dynamics. This contradicts the definition of capability as a “potentiality”, i.e., a potential behavior, not the behavior itself. Furthermore, the ArchiMate metamodel does not include elements related to “aspect” or “characteristic”, which we call issue I_1 .

Analysis of capability meaning As mentioned, the ontological analysis of the capability meaning related to issue I_1 was already addressed in [4] and here we will delve into it. As addressed, UFO embraces ArchiMate’s fundamental distinctions. However, one fundamental difference between UFO distinctions and the ArchiMate metamodel, that impacts directly this analysis, concerns the UFO-aspect distinction. UFO regards aspects as intrinsic characteristics of *endurants* (especially objects). And, as mentioned above, UFO concerns dispositions, a specialization of aspects that are manifested through events. Based on these ontological distinctions, as approached in [4], capabilities do not correspond to behaviors (events), or they do not correspond to structure (objects); instead, they are aspects (dispositions) that characterize objects (structure) manifest through events (behavior).

3.2 Capability Composition and Decomposition

When modeling capabilities in activities such as capability mapping, they are structured hierarchically in order to better understand them and support the gap analysis. E.g., in Figure 2, “Sales and Distribution” is decomposed in “Distribution Channel Management” and “Sales Execution”. There are generally two mapping approaches, top-down and bottom-up [46]. In this context, the decomposition (and composition) of capabilities can be a challenging task. They can just represent the detailing of directly capabilities of a same entity (e.g., the whole organization) into simple capabilities of the same entity; or, it can involve complex phenomena, such as the phenomenon of emergence (e.g., as happens with capabilities such as resilience, adaptability, innovation, etc). In this case, the decomposition does not consider just the whole organization, but also its parts and how they are connected. So, the capability of a whole (e.g., organization) is detailed in the capabilities of parts (e.g., organizational units, teams, or employees), which can be detailed more, and so on. The decomposition of capabilities in the second case is not as straightforward as the decomposition in the first case due to the emergence phenomenon. Capabilities are inherently more difficult to decompose due to their nature.

ArchiMate Issues In ArchiMate, the composition and decomposition of capabilities are the most common representation, as depicted in Figure 2. They are represented using structural relationships, specifically aggregation and composition relationships. These relationships indicate that a capability consists of one or more capabilities. However, in the composition relationship, there is an existential dependency between the whole and its parts. Notably, ArchiMate’s aggregation and composition relationships do not consider the *emergent capabilities* and their distinction with *complex capabilities*; they merely relate a capability to its parts without making this distinction explicit. Consequently, there are no guidelines about how to deal with the emergence of capabilities. We call the lack of emergence distinction of issue I_2 and lack of complex capability distinction of issue I_3 .

Analyzing the Capability Decomposition According to ontological distinctions, the capability of an object can be complex capability or simple capability. The complex capability of a bearer is composed by other capabilities of the same bear. In this case, the capability decomposition is not related to wholes and their parts. Also based on ontology, capabilities can also be classified as an emergent or resultant capability when characterizing a whole or system (in a broader sense). In this case, the capability decomposition must follow the structure of the whole and its parts. In this case, the emergent capability is a capability of the whole that arises from the combination of interactive capabilities of the parts (capabilities that have interactive relationships). As a result, based on the ontology, capability decomposition can have two distinct meanings: (i) *capability detailing*: when a complex capability is just split into simple capabilities independently of the system structure (addressing issue I_3); and, (ii) *capability partitioning*: when a system capability (capability of a whole), which can be emergent or resultant, is described based on the particular capabilities, i.e., capabilities of system components (addressing issue I_2). The capability ontology not only identifies these distinctions but also emphasizes that, as important as decomposing an emergent capability into particular capabilities, is understanding the system composition (how the bearer is broken down) and how the particular capabilities interact with each other.

3.3 Interaction Between Capabilities

Another issue related to EA and capability modeling is related to capability interaction. Capabilities are not isolated entities since they can be combined, impact, and collaborate with each other. Sometimes, they depend on each other to satisfies a goal. E.g., in Figure 2, “Campaign Management” needs the outcomes from “Marketing Development” in order to works; the same happens between “Relation Management” and “Contact Managment”. In the context of EA, it is important to identify how capabilities impact enterprise elements, especially other capabilities [46, 47]. For instance, initiatives to increase the innovation capability in organization can generate unexpected side effects. This, in the end, can harm other capabilities from a global perspective of the organization, as [41] argues about organizational learning. A similar issue happens with technical and social capabilities in organizations as socio-technical systems [3]. For example, high investments are made in organizations to acquire new technical capabilities, but without having a clear idea of the undesired impacts of such new capabilities on the whole

organization, especially in social ones [15]. This can lead to losses in many aspects [37]. In a way, this issue is related to the previous one, as it also involves understanding how capabilities interact in order to know how they can be combined.

ArchiMate Issues To help with understanding capabilities, ArchiMate allows the representation of dynamic and dependency relationships between capabilities. However, these relationships are not so common as the hierarchical decomposition in capability models, as happened in the official case study [20]. Besides this, capabilities can serve, trigger, or flow to other capabilities, as mentioned before. The serving relationship is a dependency relationship that indicates a “contribution” from one capability to another. The triggering relationship is dynamic, representing temporal or “causal precedence” between capabilities. The flowing relationship is also dynamic, corresponding to an “exchange” of matter, energy, or information between capabilities. These definitions in ArchiMate do not consider the capability nature and are not precise (issue I_4), which leads to a semantic overlapping (issue I_5). For example, the meaning of the serving relationship between capabilities in ArchiMate, defined as “one capability contributing to another”, is unclear and somewhat cyclic, as “contributing” is very similar to “serving” and is quite vague as well. As a result, the meaning of serving can overlap the flowing and triggering relationships. Besides this, these definitions of capability relationship in the ArchiMate specification are defined from the perspective of behavioral elements, as if capabilities were directly comparable to events and processes which have temporal precedence and can exchange matter, energy, or information. However, as we have discussed before, capabilities are to be distinguished from their manifestations, the latter of which are events.

Analyzing the dynamic relationships between capabilities. Analyzing the dynamic relationships (flowing and triggering) based on ontology, we understand that capabilities can interact indirectly through their manifestations. Using ontology, we can better grasp the semantics of flowing and triggering relationships (addressing issue I_4). Ontologically, the flowing relationship means that capability c_1 manifests through an event e_1 , producing output o_1 . After this, capability c_2 manifests through event e_2 , using output o_1 as input. Thus, it is the manifestation e_1 that flows to manifestation e_2 in the end. For the triggering relationship, capability c_1 triggering capability c_2 means that c_1 manifests through event e_1 , generating an outcome o_1 . This outcome activates capability c_2 , triggering its manifestation through event e_2 . Finally, besides triggering relationships, the ontology also provides the distinction of blocking (or disabling) capabilities, not addressed by ArchiMate.

Analyzing the serving relationship between capabilities. Based on ontology, we can interpret the term “contribute” used in serving relationship definition in various ways, leading to different meanings that need to be addressed. Depending on the interpretation, there can even be an overlap between the serving relationship and other relationships. First, based on ontology, serving could mean that capability c_1 serves capability c_2 if c_1 provides something (matter, information, or energy) to c_2 . However, this overlaps with the flowing relationship, where one capability provides something to another. Second, serving could mean “contributing to the manifestation”, where c_1 serves c_2

if c_1 contributes to the manifestation of c_2 . This interpretation also overlaps with the flowing relationship, as it represents the creation of conditions for the manifestation of another capability (the first and second meaning must be not considered to avoid semantic overlapping, addressing issue I_5). Third, a similar interpretation involves reciprocity. Ontologically, reciprocal capabilities need each other to manifest together (e.g., Product Purchasing and Product Selling). Here, c_1 serves c_2 if they are reciprocal and contribute to each other's manifestation. Fourth, serving could mean "additionality", i.e., capabilities c_1 and c_2 manifesting through e_1 and e_2 events and contributing with the same outcome o_1 . For example, the capability of software testing contributes to the software coding capability by improving the quality of the software coding output. In capability ontology, these capabilities are called "additional". Fifth, serving could mean that c_1 contributes to the development (improvement) of c_2 . Ontologically, a capability can change another; for instance, the manifestation event e_1 of capability c_1 changes the capability bearer b_1 characterized by the capability c_2 , consequently also changing c_2 . In summary, the serving relationship between capabilities has semantic overload and also overlap. So, it need to be distinguished to triggering or flowing relationships (addressing issue I_5) and need to be specialized in reciprocity, additionality, or changing relationships (addressing issue I_4).

3.4 Capabilities and Resources Assignment

Organizations as a kind of socio-technical systems include distinct kinds of resources, from people to equipment such as hardware, software, tools, and physical environment that interact [3, 13, 28]. This interaction between resources must occur appropriately for the organization to develop desirable capabilities. However, understanding how resources interact can be an intricate issue since these distinct kinds of resources (especially human and technical ones) have distinct natures [23]. For example, it is common to see in organizations the idea that the acquisition of technical resources, e.g., a new software or tool, will solve all problems and make the whole organization develop and achieve a desired capability. However, in this example, if people do not have the correct skills to properly use such technologies or have the incorrect attitude (e.g., resistance to technologies), such investment will be fruitless [3, 15]. So, the acquisition of new resources (in a not integrated way) does not imply improvements in the people and organization as a whole [3, 8, 13].

ArchiMate issues In ArchiMate, resources are structure elements that belong to the strategy layer. According to the specification, resources include *tangible assets* (i.e., physical or financial assets), *intangible assets* (i.e., technological, reputational, or cultural assets), and *human assets* (e.g., skills, knowledge, or know-how). This presents a contradiction, as this definition implies that intangible assets, as skill or competences, also can be structure elements, i.e., part of organizational structure in a similar way to organizational units or equipment. However, concerning ArchiMate metamodel distinctions, they are more similar to capabilities. Again, this issue is related to the lack of an element to represent aspects in ArchiMate, as mentioned earlier. As ArchiMate do not have it, and the focus here is to analyze capability and its relationships, we will focus solely on tangible assets in the assignment relationships. According to the ArchiMate metamodel,

resources can be assigned to capabilities. The assignment relationship is structural and it relates structure elements to behavior elements, indicating the “allocation” of structure elements to a behavior. This relationship is typically used to link business actors to a business role or business process they will perform. However, the exact meaning of the assignment relationship between resources and capabilities is not explored in detail. Specifically, it is unclear what the allocation of a resource to a capability entails (issue I_6). In an organizational context, it is common to allocate resources to a project, team, or initiative, but not to a capability. Additionally, a capability can have many resources assigned to it, raising questions about how these resources participate in that capability achievement or how they should be combined or related. As mentioned, capabilities are a result of a combination of resources. However, ArchiMate does not provide guidelines on how to combine these resources (issue I_7). Additionally, ArchiMate does not allow for the explicit representation that a resource has a certain capability; it only allows for resources to be assigned to the capability (issue I_8). To effectively combine resources in order to achieve a certain capability, it is necessary to understand their nature and also their dispositions (including capabilities and vulnerabilities).

Analyzing the resource assignment to capabilities Based on the ontology, we can imply that (tangible) resources are objects in UFO. The capability concept has distinct relationships with objects in the capability ontology. Specifically, they are related to a *capable object*, the bearer of the capability, and they are indirectly related to *capability resources*, an object needed for the capability manifestation. Thus, based on these ontological distinctions, a resource assigned to a capability can have two meanings (addressing issue I_6): (i) resource r_1 is assigned to capability c_1 if it corresponds to a *capable object* characterized by a capability c_2 , one of same type of c_1 (e.g., the software developer human resource assigned to the software development capability); (ii) resource r_1 is assigned to capability c_1 if it correspond to the *capability resource* characterized by the capability c_2 and needed to the capability c_1 manifestation (e.g., the laptop equipment assigned to the software development capability). This understanding aligns with the perspective presented in [4]. As addressed in the ontological analysis [4], to be assigned to a capability, (tangible) resources must have dispositions aligned with this capability. However, as noted in [4], this mapping is not one-to-one as in the examples presented, and often, multiple resources are assigned to the same capability. For example, the *software development capability* may involve various human resources (front-end developer, back-end developer) as well as other tangible resources like hardware and software. In this case, based on the capability ontology to address issue I_7 , all these resources assigned to a capability c are objects o_1, o_2, \dots, o_n characterized by respective capabilities c_1, c_2, \dots, c_n which interact to each other (i.e., reciprocal, additional, enabling, changing, and so on). As a result, these capabilities combined are responsible for achieving a capability c' , a capability of the same type of the desired capability c . Finally, resources in the ontology can also be capable objects and have capabilities and other dispositions (issue I_8).

Analyzing resources assignment from a system's perspective To further deepen the analysis related to issue I_7 , we can consider the distinctions of the system ontology [11]. As mentioned in the ArchiMate specification, the assignment relationship alludes to the

allocation of resources in a capability. Based on this meaning, we can imply that when a resource is assigned to a capability this means that it is allocated to a *role* (function or position) related to this capability in the organization (or its parts). Concerning the distinctions of the system ontology, we can understand the organization and its parts as a specialization of *system*, formed by interconnected *components*. In ontology, each component of a system has a *functional role* on it. Regarding the resources, its “assignment” can be understood as the allocation to a *functional role* in the *organization system* (or subsystem) in order to contribute to the achievement of the referred capability. As a result, we can conclude that the assignment of resources r_1, r_2, r_n to capability c in the organization o_1 means that these resources are a needed *components* of the organization o_1 as a system in order to achieve capability c' , one of the same type than c . In this understanding, the referred resources, as a components of the organization, must have respectively capabilities c_1, c_2, \dots, c_n and perform distinct functional roles f_1, f_2, f_n that contributes to the achievement of capability c' . In the example of the “software development capability”, behind all distinct human resources assigned to this capability there is a (social) subsystem of the organization—a team—composed of front-end developers, back-end developers, etc., and that collaborates with the “software development capability” achievement. In this case, particularly, the assignment of these human resources to this capability means that they are allocated to a team that must have this capability.

3.5 Capability Implementation

In the EA context, capability can be approached from a strategic perspective and also from a more operational perspective. In the strategic perspective, capabilities are seen in a more abstract way, independent of the implementation. In this case, capability models are more future-driven, focusing on the desired capabilities of the organization. On the other hand, from the operational perspective of capabilities, they are represented as they are in the present, considering the actual stage of the organization. The focus is more on the present and on the “how”, not just on the “what”. From the operational perspective, understanding how capabilities manifest is important to understand how they work in practice and how to implement them. In this case, not only the manifestation of the capability itself but also how the manifestation of distinct capabilities are linked and also the results and outcomes of these capability manifestations are related. The manifestation of capability generates direct and indirect results that must be understood for the capability from the operational perspective.

ArchiMate Issues Concerning the strategy and operation perspective, ArchiMate distinguishes between the strategy and business layers. The strategy layer is more abstract and implementation-independent, while the business layer is more specific, focusing on how the organization is implemented. Besides this distinction between layers, ArchiMate do not distinguish the nature of elements from strategy and business layer. In this case, the definition of capabilities is quite general and does not distinguish between strategic and operational capabilities (issue I_9). In addition, the notation only allows for the representation of capabilities in the strategic layer, leaving no construct in the business layer to represent “operational” capabilities. ArchiMate allows relationships between

elements of these layers through the *realization relationship*. This relationship, as other capability relationships, are quite vague (issue I_{10}) and generally indicates that more abstract elements (focused on “what”) are realized by more tangible elements (“how”). Regarding capabilities, they can be realized by structure or behavior elements. Thus, an “abstract” capability can be implemented by various elements combined in the business layer, such as interrelated business actors, roles, services, processes, events, and so on. This permissiveness in the language, however, lacks guidelines on how to implement capabilities, combining these elements (issue I_{11}).

Analysis of Capability Implementation Concerning the capability ontology, the strategy and operation perspectives of capability can represent different abstract levels. In this sense, addressing the issue I_{10} , the realization relationship signifies specialization when considering *capability types*. I.e., a *concrete capability type* (those with instances) specializes an *abstract capability type* (issue I_9). For example, the *software development capability* (an *abstract capability type*) can be specialized as a *mobile app development capability* (a *concrete capability type*). Regarding individual capabilities, we can understand realization in the context of *complex capabilities*. For example, if a specific organization has many types of *software development capabilities*, such as *web system development*, *mobile app development*, *desktop software development*, we can understand that these specific capabilities are part of the first (more abstract). Based on the ontology, the realization of capabilities would only occur between capabilities (i.e., abstract capability and concrete capability types), not other kinds of elements such as objects or events. When this happens, the realization relationship is a simplification that hides other relationships and has distinct meanings. So, concerning the ontology in order to address issue I_{10} , if an object o_1 (representing a structure element) realizes capability c_1 , this means that o_1 has a capability c_2 that realizes c_1 . If an event e_1 (representing a behavior element) realizes capability c_1 , this means that there is an object o_1 with a capability c_2 that is manifested through e_1 and also realizes c_2 . Based on the ontological distinctions, it is also possible to distinguish important aspects of capability implementation. As it has distinctions such as *capable object*, *capability output*, *capability outcome*, and *capability context*, it is possible to make clear these aspects and other relevant factors to the capability implementation. In order to better understand how business elements can be “combined” to realize a capability and address the issue I_{11} , we can even consider the notion of *system* (from system ontology [11]) to guide the representation of these elements. Based on ontological distinctions, we can consider that an *abstract capability* can be implemented by capabilities of a system, formed of interrelated *components* and that is manifested by *system events* (and processes).

4 Preliminary Suggestions to ArchiMate Enhancement based on the Ontological analysis

Regarding capability (de)composition, it would be beneficial to distinguish between complex and emergent capabilities in the hierarchical representation used in capability mapping. Since the strategy view does not allow the representation of the bearer of capabilities, it would be helpful to represent different levels of organizational granularity.

At least three levels could be represented: the organizational level, the organizational unit level (e.g., teams, departments), and the individual level. Capabilities at each level could be visually distinguished (e.g., by color) to indicate this difference. Capabilities that compose or aggregate others from a higher level would be emergent capabilities, while those from the same level would be complex ones. In this case, the level of capability could be assigned by an attribute.

Concerning capability interaction, it would be useful to have guidelines to model the capability interaction viewpoint, focusing on the interaction between capabilities that belongs at the same level in the organization. This would help better understand the emergence of capabilities, which arise from their interactions. Additionally, it is important to clarify the semantics of capability relationships, as their current descriptions are generic and similar to relationships between structural elements, which have a different nature. Specializing the semantics of the serving relationship between capabilities is recommended, including all possibilities such as reciprocity, additionality, and change. The serving relationship description would also make clear the distinction between it and flowing and triggering relationship in order to avoid semantic overlapping.

Regarding the representation of capabilities, a general suggestion is to consider adding an element in the metamodel for aspects as a distinct element comparing to the behavior element and structure element. This might be challenging to implement as it demands a deep change in the core of the language metamodel. Another option is to broaden the meaning of behavior elements to include “potential behavior” (potentiality). This could even be represented in the language using attributes. It is important to make the definition of capability clearer. Currently, according to the specification, a capability can belong to the whole organization or its parts. However, the language contradicts the specification descriptions having restrictions and does not explicitly allow capabilities to be related to different kinds of bearers (e.g., an organizational unit or business elements). In practice, the capability construct in ArchiMate typically represents basically *business capabilities*, i.e., capabilities of the entire organization. Therefore, it should be explicitly stated that capabilities can also belong to parts of the organization and not just the whole organization.

In addition, capabilities are mostly represented abstractly in the strategy layer. The notation should also allow the representation of “concrete” capabilities in the business layer. This would enable better representation of the capability bearer, manifestation, context, inputs, outputs, and outcomes—important distinctions highlighted by the ontology. The *business function* elements could be used to represent “concrete” capabilities, as is depicted in example of Figure 2. However, their semantics would need to be adapted to be considered an ability explicitly to “concrete” capabilities rather than a process. Additionally, the language should provide guidelines on how to implement resources to achieve the capability they are assigned to, ensuring they are realized by interconnected business elements. The notion of a system is essential in this context.

5 Related Works

Other related works that employ Foundational Ontologies in EA modeling include [5,34,36,39]. Azevedo et al. [5], performs an ontological analysis of capability, resource,

and competence. The authors discuss especially the definition of capability based on UFO; and, we adopt and build up on that analysis in the present work. As an application, the author proposes improvements in Enterprise Modeling (using ArchiMate), through a metamodel connecting capabilities and the strategy layer with motivational aspects. Capabilities can be aggregated (with resources) in what the author called “capability bundles” [4], thereby connecting individual-level capabilities (competences) with organizational capabilities. The work proposed by Nardi et al. [34] focuses on the ontological analysis and modeling of the Service concept. In a complementary way, [34] states that one dimension of Service Modeling is to represent a manifestation of capabilities. However, although both papers address the subject of capabilities using UFO, they do not delve into this topic since this is not the focus of both papers. Sales et al. [39] proposed improvements in the ArchiMate notation based on an ontological analysis, focusing on the concept of value. In their analysis, they included the concept of capability, which is strongly related to this domain. According to their interpretation, and as adopted here, capabilities are dispositions with valuable impacts on a value subject. However, including capabilities was not the main focus of their work. Building on the work in the value domain proposed in Sales et al. [39], Oliveira et al. [36] conducted an ontological analysis of ArchiMate focused on security and proposed a redesign of the language. In this security-focused analysis, they considered other types of dispositions, such as vulnerabilities, corresponding to those with undesired effects. Additionally, the authors extended the meaning of capabilities to include threat capabilities.

6 Final Remarks

In this work, we presented an ontological analysis of ArchiMate based on a well-founded capability ontology. To perform this analysis, we first identified several issues concerning the semantics of the ArchiMate metamodel, especially regarding capability emergence and capability interaction phenomena. We also uncovered semantic issues related to the relationship between capabilities and resources, as well as capability implementation. Based on these issues, we performed the ontological analysis and clarified some distinctions. Finally, we proposed improvement suggestions based on the ontological analysis.

This work impacts the improvement of capability modeling using ArchiMate and can serve as a foundation for proposing language redesign, language patterns, or even language extensions. The suggested enhancements can be a starting point for the proposal of new capability representations. This work not only contributes to enhancing ArchiMate’s capability modeling but also impacts capability modeling in general, potentially influencing other EA notations. In future work, we intend to perform an ontological analysis of the Unified Architectural Framework [27] and propose enhancements. Additionally, we aim to refine capability representation proposals based on this ontological analysis.

References

1. 42020, I.: Software, systems and enterprise–architecture processes (2019)

2. 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)
3. Appelbaum, S.H.: Socio-technical systems theory: an intervention strategy for organizational development. *Management decision* **35**(6), 452–463 (1997)
4. 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)
5. 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>
6. 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)
7. Bernus, P.: Enterprise models for enterprise architecture and ISO9000:2000. *Annual Reviews in Control* **27**(2), 211–220 (Jan 2003)
8. Bruseberg, A., Lintern, G.: Human factors integration for modaf: Needs and solution approaches. In: INCOSE International Symposium. vol. 17, pp. 1240–1255. Wiley Online Library (2007)
9. Bunge, M.: *Treatise on Basic Philosophy. Ontology II: A World of Systems*. Springer Netherlands, Dordrecht, Netherlands (1979)
10. 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)
11. 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)
12. Choi, S., Fara, M.: Dispositions. In: Zalta, E.N. (ed.) *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Spring 2021 edn. (2021)
13. Cummings, T.G.: Self-regulating work groups: A socio-technical synthesis. *Academy of management Review* **3**(3), 625–634 (1978)
14. 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>, <https://doi.org/10.1002/sys.21383>
15. Ellen, P.S., Bearden, W.O., Sharma, S.: Resistance to technological innovations: an examination of the role of self-efficacy and performance satisfaction. *Journal of the academy of marketing science* **19**, 297–307 (1991)
16. 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)
17. 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/AO-150157>
18. Guizzardi, G.: *Ontological Foundations for Structural Conceptual Models*. No. 15 in Telematica Institute Fundamental Research Series, Telematica Instituut, Enschede, The Netherlands (2005)
19. Guizzardi, G., et al.: Towards ontological foundations for the conceptual modeling of events. In: *Conceptual Modeling*, pp. 327–341. Springer Berlin Heidelberg (2013)

20. Jonkers, H., Band, I., Quartel, D., Lankhorst, M.: Archisurance case study (version 3.1). Tech. rep., The Open Group (2019)
21. Josey, A.: TOGAF® version 9.1-A pocket guide. Van Haren (2016)
22. Juarrero, A.: Dynamics in action: Intentional behavior as a complex system. *Emergence* **2**(2), 24–57 (2000)
23. Kochan, T., Cutcher-Gershenfeld, J.: Integrating social and technical systems: lessons from the auto industry (2008)
24. Maier, J.: Abilities. In: Zalta, E.N., Nodelman, U. (eds.) *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Fall 2022 edn. (2022)
25. Martin, J.: Enterprise capability management (2021)
26. Martin, J., Faisandier, A.: Enterprise systems engineering background (2021)
27. 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)
28. Mavor, A.S., Pew, R.W.: Human-system integration in the system development process: A new look. National Academies Press (2007)
29. Merrell, E., et al.: *Capabilities* (2022)
30. Molnar, G., Bradley, N.: *Powers: A study in metaphysics*. Clarendon Press (2003)
31. Morgan, P.: The concept of capacity (draft version). Study on capacity, change and performance pp. 1–19 (2006)
32. Mossio, M., et al.: Emergence, closure and inter-level causation in biological systems. *Erkenntnis* **78**(S2), 153–178 (2013)
33. Mumford, S., Anjum, R.: *Getting Causes from Powers*. Oxford University Press (2011)
34. 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>, <https://doi.org/10.1016/j.is.2015.01.012>
35. O’Connor, T.: Emergent properties. *American Philosophical Quarterly* **31**(2), 91–104 (1994)
36. Oliveira, Í., Sales, T.P., Almeida, J.P.A., Baratella, R., Fumagalli, M., Guizzardi, G.: Ontological analysis and redesign of security modeling in archimate. In: *IFIP Working Conference on the Practice of Enterprise Modeling*. pp. 82–98. Springer (2022)
37. Rai, A., Patnayakuni, R., Patnayakuni, N.: Technology investment and business performance. *Communications of the ACM* **40**(7), 89–97 (1997)
38. Rosemann, M., Green, P., Indulska, M.: A reference methodology for conducting ontological analyses. In: *Conceptual Modeling. ER 2004*. vol. 3288, pp. 110–121. Springer, Berlin, Heidelberg (2004)
39. Sales, T.P., Roelens, B., Poels, G., Guizzardi, G., Guarino, N., Mylopoulos, J.: A pattern language for value modeling in archimate. In: *Advanced Information Systems Engineering: 31st International Conference, CAiSE 2019, Rome, Italy, June 3–7, 2019, Proceedings* 31. pp. 230–245. Springer (2019)
40. Saxena, M.: *Capability Management*. Global India Publications (2009)
41. Senge, P.M.: *The fifth discipline. Measuring Business Excellence* (1997)
42. 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>, <https://doi.org/10.1002/j.2334-5837.2013.tb03050.x>
43. Spencer-Smith, R.: Reductionism and emergent properties. In: *Proceedings of the Aristotelian Society*. pp. 113–129. JSTOR (1995)
44. 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)

45. The Open Group: Archimate 3.2 specification (2023), <https://pubs.opengroup.org/architecture/archimate3-doc/>
46. The Open Group: TOGAF Business Capabilities Guide V2. <https://pubs.opengroup.org/togaf-standard/business-architecture/business-capabilities.html> (2023), [Accessed 06-10-2023]
47. Thorn, S.: Redefining traceability in enterprise architecture and implementing the concept with togaf 9.1 and/or archimate 2.0 (2013)