Software and Systems Modeling
https://doi.org/10.1007/s10270-023-01146-w

THEME SECTION PAPER

Toward an ontology for EA modeling and EA model quality

Jan A. H. Schoonderbeek[1] · Henderik A. Proper[2]

Received: 7 April 2023 / Revised: 15 December 2023 / Accepted: 18 December 2023

Abstract
Models have long since been used, in different shapes and forms, to understand, communicate about, and (re)shape, the world around us; including many different social, economic, biological, chemical, physical, and digital aspects. This is also the case in the context of enterprise architecture (EA), where we see a wide range of models in many different shapes and forms being used as well. Researchers in EA modeling usually introduce their own lexicon, and perspective of what a model actually is, while accepting (often implicitly) the accompanying ontological commitments. Similarly, practitioners of EA modeling implicitly also commit to (different) ontologies, resulting in models that have an uncertain ontological standing. This is because, for the subject domain of enterprise architecture models (as opposed to the content of such models), no single ontology has gained major traction. As a result, studies into aspects of enterprise architecture models, such as "model quality" and "return on modeling effort", are fragmented, and cannot readily be compared or combined. This paper proposes a comprehensive applied ontology, specifically geared to enterprise architecture modeling. Ontologies represent structured knowledge about a particular subject domain. It allows for study into, and reasoning about, that subject domain. Our ontology is derived from a theory of modeling, while clarifying concepts such as "enterprise architecture model", and introduces novel concepts such as "model audience" and "model objective". Furthermore, the relevant interrelations between these different concepts are identified and defined. The resulting ontology for enterprise architecture models is represented in OntoUML, and shown to be consistent with the foundational ontology for modeling, Unified Foundational Ontology.

Keywords Enterprise architecture · Ontology · Domain model · Enterprise architecture modeling · Enterprise architecture model · Architecture · Model quality

1 Introduction

1.1 Models to understand and shape the world

In dealing with complex phenomena, such as processes in nature, the construction of a building, the design of an information system, etc, we tend to '*work with*' an *abstraction* (in our mind) of the actual phenomenon, while zooming in on those properties of the phenomenon that matter to us, and filtering out all the properties that are not relevant to the goals at hand. When we decide to externalize this *abstraction* in terms of some artifact, then this artifact is a model (to us, as an individual) of the observed phenomenon [50].

In general, the resulting models may take different shapes and forms, such as sketches, precise drawings, textual specifications, formal specifications, or tangible forms mimicking key physical properties of some original. The suitability of these different shapes and forms of models, depends on the purpose (including the audience) the model should serve [17, 51]. This general view on models is in line with foundational work on the concept of model by, e.g., Apostel [3], and Stachowiak [63]. What is important to stress here, is the fact that the notion of model should not be "*framed*" into *only* referring to formal models or boxes-and-lines diagrams.

In a development/engineering context, models can be used to, e.g., study, describe, or explain, the current state of affairs, as well as capture (at different levels of precision, detail, and normative strength) guidance toward a future desired state of affairs.

Communicated by Dominik Bork and Sybren de Kinderen.

✉ Henderik A. Proper
Henderik.Proper@tuwien.ac.at

Jan A. H. Schoonderbeek
J.Schoonderbeek@architecting.nl

[1] Maastricht University, Maastricht, The Netherlands

[2] TU Wien, Vienna, Austria

Published online: 03 February 2024

1.2 The role of models in EA

When an organization requires change, it is up to the professionals in the field of enterprise architecture (EA) to provide advice to decision makers on how to structure the organization and its different systems to enable this change, as well as guiding the implementation parties [35, ch.1]. To that end, enterprise architects capture and use different architectures, such as the baseline and target architectures, as well as relevant reference architectures [15, 72].

Depending on the needs at hand, such architectures can be captured by way of different models. In terms of [49], the involved models may follow an *extensional* style, involving explicit representations of, e.g., actors, roles, processes, and objects, as generally used in graphical modeling languages. They may also follow an *intensional* style involving, e.g., architecture principles [15] capturing general directive statements, design guidelines or design constraints.

The resulting models, in particular those that follow an *extensional* style, may be too detailed or too formalistic for a given target audience and/or purpose. In the context of the ArchiMate research project, this observation resulted [35, ch.7] in a basic framework of viewpoints (following the definitions of view and viewpoint from [25]) involving different overall purposes and target audiences. As also suggested in [35, ch.7], the resulting views may use different notations and formats. These views are, however, still models in the sense of being an externalized abstraction of some existing, future, or desired domain. This viewpoint framework is still part of the specification of the ArchiMate standard [69, ch.13], as evolved from the original ArchiMate project [35].

Constructing models/views targeting different audiences, and non-technical audiences in particular, is a major challenge in itself. In the aforementioned ArchiMate research project, this also resulted in some initial guidelines [35, ch.6]. More recently, in [61], the authors also stress, from a practitioners' perspective, the need to better match models/views to the "*languages of our target audiences*" such that "*they might, finally, get what we were trying to say all along.*"

At a more fundamental level, bridging between different audiences (e.g., between IT architects, Business Architects, and business stakeholders) is related to the concept of boundary objects. Griesemer and Star [64] coined the notion of boundary objects as a bridge between different social worlds: "*They have different meanings in different social worlds but their structure is common enough to more than one world to make them recognizable; a means of translation*" [64].

The concept of boundary objects was used in [48] to study the role of architectural descriptions in the context of software architecture, while in, e.g., it was investigated in an enterprise architecture context [1, 2, 5]. In [33], the authors provide a broader overview of the role of different enterprise architecture artifacts as boundary objects. In line with our broad understanding of what a model is (which, in the context of EA, we will elaborate in more detail upon in Sect. 4), we would argue that in an EA context, most artifacts actually involve models. This view is strengthened by the analysis as reported in [21], involving an empirical study into EA artifacts. This latter study identifies 43 types of EA artifacts. The definition of these artifacts frequently refer to notions such as: *roadmap*, shared *view*, (heat)*maps*, *trees*, *structure*, and *wireframes*, which, so we would argue, are essentially all manifestations of models.

1.3 The problem of general EA model quality

As models are contained in, or provide an underpinning of, many of the architects' professional products and services [33, 36], it becomes important to consider the needed qualities in relation to their context of use. As discussed above, these qualities involve different aspects, including requirements on, e.g., the form they take, the level of specificity or formality, etc.

Much work has been done on the quality of visualization [74] for EA models. But where the quality of the model itself is concerned, there is little theory on obtaining/maintaining sufficient quality [71] (notwithstanding works like [34, 37]). Even practical guidance based on scientific research, such as [36, ch.7], seems to be thin on the ground[1]. Thus, the creation of EA models must currently be considered a best-effort affair, for which the quality of its results cannot be objectively determined. More research into EA model quality is clearly needed, and in this paper, we set out to provide a solid foundation for such research.

1.4 An ontology for research into EA model quality

As argued in [8], conducting research requires a suitable conceptual framework that defines:

- Epistemology—*Why is the research required or desired?*
- Ontology—*What is it is that is being researched?*
- Methodology—*What are the precise research questions, and how will the research be conducted?*

For EA model quality, the general context for why more research into this topic is needed, is outlined in Sect. 1.3. In the remainder of this paper, we focus on the ontology for research into EA model quality. In doing so, we will follow the following definition of ontology:

[1] As an illustration, a search on Google Scholar combining the three terms `"practical guidance"`, `"modeling"`, and `"enterprise architecture"` (including the quotation marks) returned 273 results. Of these, none had as its main topic the practical guidance for EA *modeling* itself.

> "An ontology is a conceptual model of (a fragment of) an observed reality; it is, in essence, a repository of interlinked concepts pertaining to a given application domain." [12, p.79]

We are not alone in noting that for research into *EA model quality*, no suitable ontology readily presents itself:

> "... we identify a lack of work regarding quality assessment of enterprise architecture models in general and frameworks or methods on that account in particular." [71, p.14]

Many terms that are directly or indirectly involved with EA model quality have no definition that is widely agreed upon in the communities of researchers, nor of EA practitioners. Even the principal terms *EA model*, and *model* itself, are not well defined.[2] As a further complication, polysemy and homonymy [27, 32] present a challenge, as several terms to do with EA models have either of these characteristics, including such terms as *capability*, *concept*, *domain*, *model*, and *scope*. For such a term it may not be automatically clear which one of its definitions pertains. Furthermore, for some of these terms the different definitions themselves are not even settled.

Problems arising from lacking, imprecise, or otherwise impacted terms and concepts surrounding enterprise architecture model quality are not limited to the field of research. Practitioners who create and work with enterprise architecture models are also impacted. As it stands, enterprise architecture modeling seems mostly a "black art", whereby modelers lack practical guidance on how to arrive at a sufficiently "good" enterprise architecture model [37, p.42], [44, p.202], [71, p.14]. At the basis of this problem lies the lack of a suitable ontology with which to express modeling, model results, and model quality.

In summary: both scientific investigation into enterprise architecture model quality, and attaining required levels of enterprise architecture model quality in practice, are hampered by the fact that the relevant terms, concepts, and their interrelation are ill defined. This problem will be addressed in this paper by answering the following question: what constitutes a suitable ontology for investigating enterprise architecture model quality?

1.5 Structure of the paper

The structure for the remainder of this paper, which is a continuation of the research that was started in [60], is as follows:

- In the section "Research method", we describe the research approach used to develop the ontology as presented in this paper, and how we validate our results.
- The next section on "Background and conventions" provides our epistemic stance; outlines our approach to creating and validating definitions; describes the conventions for naming terms and relations; provides a small primer for the Unified Foundational Ontology (UFO); sketches how we capture the ontology in an OntoUML model; and has a section on the meaning of purpose, goal, and objective.
- Section "A definition for EA model" deals with expanding and specializing the definition for *conceptual model* into a definition for an enterprise architecture model, as well as a rudimentary theory of modeling.
- Section "Domain Lexicon" then uses all that went before to create a single model of twenty terms and forty-one relations, that form the desired ontology for EA model quality.
- The validity of the uncovered ontology is addressed in section "Validity".
- We conclude with a short summary of the results, some words on the value of the ontology, a critical evaluation, and an overview of further work.

2 Research method

2.1 Research approach and results

Ontology development is generally considered to be an engineering activity [30, ch.2]. It needs to start with establishing both a scope for the ontology, as well as criteria with which to determine when the ontology development is done. For this, [30, ch.3] advocate a use case-based approach, whereby a collection of use cases determine the scope and content of the ontology. Unfortunately, the discipline of EA and the field of EA modeling are too immature to have a lexicon that is both comprehensive and generally accepted. Thus, surveying subject matter experts for terms and definitions pertaining to EA modeling is not considered a viable approach.

There also is no suitable theory of Enterprise Architecture modeling available that can serve as a basis to derive terms and definitions that factor in the creation and exploitation of EA models. Nor is there a generally accepted lexicon or the-

[2] As an illustration, consider the systematic literature review by De Meyer and Claes [11] for one subdomain of EA models, namely business process models. This review presents 42 relevant secondary studies, containing 39 different quality dimensions and 21 quality metrics. But notably these studies did not agree, nor converge, on a single definition of process model quality, nor of model quality itself. Or consider ISO/IEC/IEEE 42010:2022(E) [28], a standard for software architecture descriptions that is well referenced by EA frameworks such as TOGAF [70] as the conceptual model for enterprise architecture descriptions in general, and consequentially also for EA models. While acknowledging the activity of modeling, this standard omits the definition of *model* (EA, architecture, or otherwise) altogether; nor does it define the activity of *modeling*.

ory of modeling for the wider domain of conceptual models, even though there is no shortage of candidates, such as [23, 39, 43].

To arrive at an initial ontology for EA modeling, we attempt in this paper to formulate a suitable theory of modeling. For this we follow these steps:

- We start from the definition of domain model, as we have been using it until now [51]. Making use of the analytical method of definition [42], we then arrive at an expanded definition for "domain model".
- We clarify the characteristics of the definition for "domain model" using existing theories such as Concept Theory [10].
- We then continue by refining the expanded definition for "domain model" into a definition for "EA models", with properly defined essential characteristics.
- From this, we can arrive at properly defined concepts and relations, necessary for the desired ontology.
- Finally, we express the uncovered concepts and relations in OntoUML (see Sect. 3.5).

For both the evaluation of existing definitions and the drafting of new definitions, we follow the methods provided by ISO704-2022 [29, ch.6], augmented with definition specifications from Saenz et al. [57, ch.2].

The result of this approach is the sought-after ontology, consisting of a conceptual model that includes both the activity of EA modeling and its result (the EA model), plus an accompanying glossary.

2.2 Validating the results

The validity of an ontology can be checked on two fronts: content validity, and application validity [65]. In order to validate the resulting ontology, we currently limit ourselves to content validity:

- While creating the ontology, we keep in mind the contents of the OntOlogy Pitfall Scanner! (OOPS!),[3] which is a "Catalogue of common pitfalls" for ontology work. This provides the first step in ensuring content validity.
- We perform an ontological analysis by checking the syntax of our OntoUML representation of our ontology. This constitutes a second step in content validity.

Any ontology is always a living document, subject to maintenance and evolution [30]. The validation phase thus is not meant to show that our ontology is "finished", but rather that it is of sufficient completeness and quality to be used and expanded in practice.

[3] The online link for OOPS! is https://oops.linkeddata.es/catalogue.jsp.

Table 1 Six criteria for judging the quality of a definition [57, p.488]

1. A definition must state essential attributes
2. A definition must be non-circular
3. A definition must be accurately scoped
4. A definition should have clarity
5. A definition should be affirmative
6. A definition should be simple.

3 Backgrounds and conventions

3.1 Epistemic stance

When considering models and their relation to the world, we take the stance of critical realism [41]. This entails that we believe there is an inherently intransitive domain of items, objects, events and such, independent of human observers (conform the philosophic stance of realism). By contrast, our knowledge of this domain is entirely dependent on us as observers and thinkers; knowledge exists in a transitive domain. But while knowledge and observation is thus epistemically relative, we do not hold that views and judgments are equally relative; our stance is critical in the Kantian sense.

3.2 Creating definitions

A major part of any ontology consists of determining formal names and definitions for the knowledge domain under consideration. In this paper we will create new definitions as intensional definitions [29], using the analytical method [42] if a definition for a superordinate concept is available. Furthermore, we will qualitatively judge new and extant definitions on the criteria from [57], listed below in table 1. In this paper, we will refer to these as *the criteria for definitions*.

When creating a definition, we will use the form as recommended by [29, ch.6.4]. This entails starting with the definiendum and a colon, then the definiens. Every essential attribute in the definiens will get its own definition, or will have been defined previously, unless it can be considered part of the background knowledge of the reader.

3.3 Conventions for naming terms and relations

We will attempt to express every term of the ontology as a noun, and then every relation between these terms as a verb, for which we will provide the present simple form. We distinguish between relations and relationships, whereby the concept of relationship is the truthmaker for the relation.[4] The relationships that describe the relation will be labeled

[4] An explanation of the subtle difference between relation and relationship can be found in [16].

with a noun that corresponds with the verb of the associated relation. To make the lexicon maximally clear and avoid ambiguity (where feasible), we will avoid using polysemic or homonymic words and verbs (as mentioned in Sect. 1). This includes avoiding the use of the same noun or verb for more than one term, relation, or relationship, as that effectively introduces polysemy. We also avoid words and verbs that are so generic and common as to not readily be recognizable as a specific term, such as *has* and *is*. Exceptions to the conventions provided above are the following special cases:

- *Has* will be used for all relations between some term that refers to something that has attributes and/or properties, and those attributes or properties. This is because we feel it would be very contrived to say something other than that something *has* that attribute or property. Furthermore, *has* is a fitting word that describes the relation between a composing or aggregating concept, and the other concept(s) that it is composing or aggregating.
- *Aggregates* will be used for every aggregation.
- *Composes* will be used for every composition.

3.4 UFO as a foundational ontology

In order to create an ontology, it is good practice to start from a foundational ontology [30, ch.5]. Such a foundational ontology provides terms to describe metadata, basic knowledge such as time, and core domain-level content. A foundational ontology specifically suitable to ground our ontology for EA modeling quality is the Unified Foundational Ontology (UFO) [19]. This ontology has extensive theoretical grounding, and offers ontological distinctions and axioms specifically geared to structural conceptual modeling [18].

Here we provide a subset of ontological distinctions that are put forward in UFO, and will be used later in this paper to express the ontology for EA model quality. In the taxonomy that UFO presents [20, p.5], types of things can be either of the class ⟨⟨Endurant type⟩⟩ (objects, with both essential and accidental properties, which may change with time) or ⟨⟨Perdurant type⟩⟩ (events and processes). Some classes of objects under the ⟨⟨Endurant type⟩⟩ are:

- ⟨⟨Kind⟩⟩ refers to a genuinely fundamental type of object that exists in a given domain. An object of a certain ⟨⟨Kind⟩⟩ cannot also belong to another ⟨⟨Kind⟩⟩, and also cannot change (or be changed) into another ⟨⟨Kind⟩⟩ without having fundamentally (being) changed itself. Examples in the domain of EA modeling are Actors, Goals, and EA Models.
- ⟨⟨Role⟩⟩ represents a specific set of properties that an object or entity in a relational context. "Being a modeler" means the entity has the intrinsic properties of being able to create models.
- ⟨⟨Collective⟩⟩ represents a plural entity, which aggregates entities as member of the collective, each of which exerts an identical role with respect to that plural entity. When an entity is part of some collective "audience", then it exerts the role of "audience member"; all other entities in that collective exert the same role.
- ⟨⟨Quality⟩⟩ refers to a particularized property, something that existentially depends on some individual ⟨⟨Endurant⟩⟩. They are reifications of categorical properties that can be either one-dimensional (properties such as size or recency) to multidimensional compositions where the value of the quality can fall within a multidimensional space, dubbed a quality domain - examples are the vowel space and the CIE 1931 color space.

Relations are entities that connect and bind together other entities. A broad category of relations is that of the ⟨⟨material⟩⟩ relations. They are called "material" because they have a material structure on their own. Every material relation has some ⟨⟨relator⟩⟩, a cluster of relational properties that mediates the relation; it acts as the truthmaker for that relation [16]. An example of a relator and its associated material relation is the "conception"; it is the relator for the relation "conceives" that holds between some person and the concept that they are conceiving. Another example is the "construction" as the relator for the relation "constructing" that holds between the role of modeler, and the model that the modeler is actually constructing.

3.5 Presenting the ontology in OntoUML

OntoUML is a general conceptual modeling and ontology representation language whose metamodel complies with UFO, so as to provide an ontologically well-founded modeling language [19]. It provides the ontological distinctions from UFO as modeling primitives. If an ontology can be encoded in OntoUML in a syntactically valid way, then this guarantees that it aligns with the UFO foundational ontology [7, 20]. OntoUML is itself an extension of the Universal Modeling Language (UML) [62].

The ontology for EA model quality will be captured in a single OntoUML model, which in turn will be presented in a number of separate views. In these views, each concept will be captured as an OntoUML class that represents an UFO class of objects, such as ⟨⟨kind⟩⟩—see previous Sect. 3.4. The relations between these will in most cases be ⟨⟨material⟩⟩ relations. Each of these will be associated with a ⟨⟨relator⟩⟩ concept, which itself is connected to the two concepts at the end points of the relation via two ⟨⟨mediation⟩⟩ relations. Further relation types that will be required are ⟨⟨componentOf⟩⟩ relations, ⟨⟨specialization⟩⟩ relations, and ⟨⟨characterization⟩⟩ relations.

The views will also present the multiplicity of each relation, as defined in UML. Multiplicity indicates how many instances of one class can be considered connected to an instance of another class through a given relation, and vice versa. This relation is presented as a string that indicates the lower and upper bounds at the endpoints of the relation. A detailed analysis of the meaning and implication of each of the provided multiplicities is beyond the scope of this paper, and will be presented in future work.

3.6 Purpose, goal, and objective

Part of the discussion of models is the term *purpose*; and we have ourselves used this term before (for example in [17]). But in everyday speech, this word purpose is not used consistently; it is often used interchangeably with goal and objective. As an illustration, consider some dictionary definitions for "purpose" [53, 54], "goal" [13, 14], and "objective" [46, 47]. These show how the colloquial meaning of the three words spill into each other, making each (almost, although not quite) a synonym for the others.

To avoid any confusion when using any of these words, it is necessary to assign to each some unambiguous, non-synonymous meaning. As dictionaries cannot provide such meaning, this paper needs to explicitly posit and use suitable definitions. We will base such definitions on the Business Motivation Model (BMM) [10], TOGAF 10th edition [68], as well as on the dictionary entries given above. Making the definitions somewhat more generic than the BMM's scope, we arrive at the following:

Definition 1 **Purpose**: the motive or reason that an actor holds, that describes why it is or should be the case that a goal is pursued, an object exists, or an action is carried out.

Definition 2 **Goal**: a statement about a state or condition to be brought about or sustained through appropriate means. (Based on [10, ch.8.2.4])

In the latter definition, the term "means" may require elucidation. From [10, ch.8.3] we get:

Definition 3 **Means**: any device, capability, regime, technique, restriction, agency, instrument, or method that may be called upon, activated, or enforced to achieve ends.

Definition 4 **Objective**: a statement of an attainable, time-targeted, and measurable result that is sought by an actor in order to achieve a goal. (Based on [10, ch.8.2.5])

For a definition of actor, we turn to TOGAF 10th edition [68, ch.4.2]:

Definition 5 **Actor**: A person, organization, or system that has one or more roles that initiates or interacts with activities.

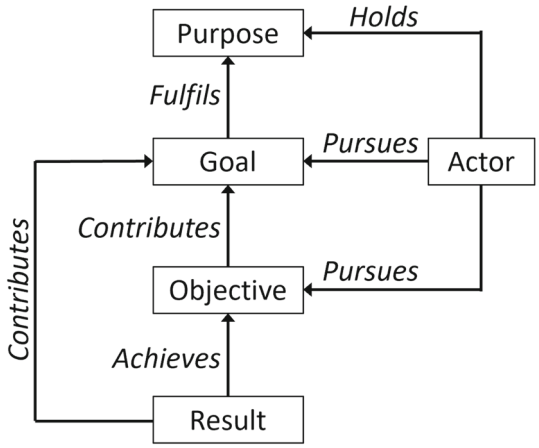

Fig. 1 The relations between actor, purpose, goal, objective, and result

One should note that purpose, goal, and objective all exist in the human mind, be it a single person or a collection thereof. Thus, if the actor in definition 1, 2, or 4 is actually a system, then the purpose, goal, or objective necessarily resides in one or more humans that employ that specific system, rather than the system itself.

The causal link to the rest of the world is played by the *result* mentioned in definition 4. Within the context of purpose, goal, and objective, we define the term result as:

Definition 6 **Result**: an outcome or situation that exists at the conclusion of some action.

If a given actor pursues a certain objective, it is likely that they will also pursue some result that achieves this objective. Nevertheless, definition 6 is not explicitly tied to any actor. This is because it could be the case that a result is obtained indirectly, or by some uninvolved actor, or even by random chance.

Definitions 1 through 6 validate well against the criteria for definitions, and allow for unambiguously identifying and labeling different useful concepts and their interrelations. On these definitions, an actor can hold some purpose (a motive), to which they then formulate and pursue a goal (the end state they wish to achieve) which will fulfill that purpose. They can then strive toward that goal by pursuing one or more objectives that each contribute to that goal. That objective can in turn be achieved in the form of a result, which is itself thus considered a contribution toward the goal. A diagram showing these terms and relations is given in Fig. 1.

Note: other publications are likely to use these terms and relations in a way that has a differing meaning. Therefore, if any of these terms appear in some source to be used, then it requires analysis, and sometimes remapping according to the definitions in this section, before the content of that source can be processed within the context of this paper.

4 A definition for EA Model

4.1 A definition for "domain model"

The analytical method described by [42] provides a means to arrive at a definition of some term, by departing from a definition for some superordinate concept. For the term Enterprise Architecture Model (EAM),[5] the concept of a "domain model" presents a suitable superordinate concept. The characteristics of this concept will carry over into the definition of an EAM, while the specific circumstances and uses of an EAM will introduce new or specialized characteristics.

Based on foundational work by, e.g., Apostel [4] and Stachowiak [63], more recent work on the same by different authors [22, 56, 58, 67], as well as our own work [9, 51, 52], we currently understand a domain model to be:

Definition 7 Domain model: a social artifact that is understood, and acknowledged, by a collective human agent to represent an abstraction of some domain for a particular cognitive purpose.

In [51] we have described the essential attributes of definition 7 as follows:

- *Social artifact*: A model is seen as a social artifact in the sense that its role as a model should be recognizable by a collective agent (e.g., people).
- *Collective Agent*: The collective agent observes the domain by way of their senses and/or by way of (collective) self-reflection, and, based on this, should acknowledge/accept the artifact as indeed being a model of the domain (for a given purpose).
- *Abstraction*: a model is the representation of an abstraction. This implies that, in line with the cognitive purpose of the model, some (if not most) details of the domain are consciously filtered out.
- *Domain*: With domain, we refer to 'anything' that one can speak and/or reflect about, i.e., the domain of interest. As such, domain simply refers to 'that what is being modeled'.
- *Cognitive purpose*: A model must always be created for some cognitive purpose, i.e., to express, specify, learn about, or experience, knowledge regarding the modeled domain. As a direct corollary to this, one can conclude that a model, being a social artifact must therefore be a language utterance, as such implying it to be a social-linguistic artifact

[5] Note that where this paper uses the abbreviation EAM, the meaning is always "enterprise architecture model", never "enterprise architecture management".

Definition 7 satisfies the criteria for definitions from Sect. 3.2, however the essential attributes are defined somewhat terse. Clarifying and elaborating these essential attributes will serve to improve and expand definition 7. This will be the goal of the next section.

4.2 Augmented definitions for the essential attributes of a "domain model"

A domain model serves a goal, and therefore has an audience

In definition 7 we have employed the essential attribute of *cognitive purpose*. However, we have argued in Sect. 3.6 that purpose (like goal and objective) is held by an actor (Sect. 3.6), therefore it cannot be that the domain model itself intrinsically *has* that purpose. We now clarify that *cognitive purpose* in definition 7 has been used as a shorthand form of: the domain model is a result that is intended to contribute to a (specifically cognitive) goal, held by some actor or actors. But it is not sufficient to refer to these as the collective agent in definition 7, as the goals of the actors *using* the domain model likely not (fully) align with those of the actors *creating* the domain model.

Making use of the definitions from Sect. 3.6, we will now introduce a distinction between those who create domain models, and those that make use of them. We label the actor or collection of actors that make use of the domain model the *model audience*, with the individual actors designated *model consumers*. Thus a model audience is a collection of one or more model consumers, whose goals the domain model is directly intended to contribute to.

Note that modelers themselves are (by definition) model consumers as well. In the modeling process, they necessarily consume the model; while modeling, they need to check if the model turns out the way they intend [52, Fig. 5]. Compare this with text writers that read their writings back to see if the result matches their intent. Furthermore, people sometimes create models just for their own purposes, for example to analyze or understand something for themselves. Thus, the *first* member of a model audience must always be the modeler themselves. Also note that a domain model may be created and used to contribute to goals of actors that are expected to directly consume the domain model, but also actors that will never interact with it. We will currently not take into consideration this second group.

For actors to be motivated to actually interact with the domain model, they must believe that this will result in some contribution to their goals (which we signaled in definition 7 by employing the term *acknowledge*). To be able to distinguish between those that directly interact with domain models and those that benefit indirectly, we now introduce the following definition:

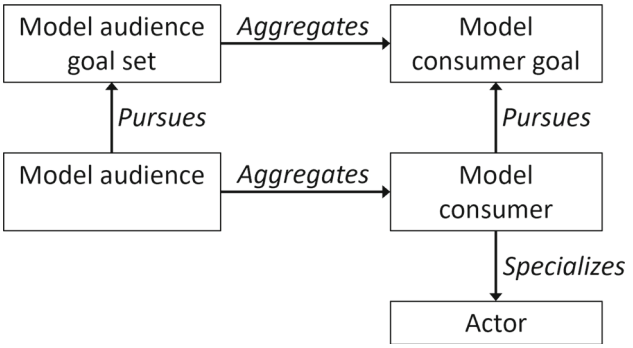

Fig. 2 The audience for a given domain model and their respective goals

Definition 8 Consuming: the action, performed by an actor, of directly interacting with a domain model (without fundamentally altering the model), independent of other actors or their assistance, on the belief that this contributes to some cognitive goal of that actor.

Thus, a member of a model audience is an actor who is expected by the modeler to *consume* the domain model.

Given this line of reasoning, and given the definitions of actor and goal as provided in definitions 5 and 2, we now come to the following definitions:

Definition 9 Model consumer: an actor who is expected to consume a domain model, on the belief that this interaction contributes to some cognitive goal of theirs.

Definition 10 Model consumer goal: a cognitive goal, pursued by a model consumer, that a model tentatively contributes to.

Definition 11 Model audience: the aggregated set of model consumers that a domain model is intended to serve.

Definition 12 Model audience goal set: the aggregated set of the model consumer goals that a domain model is intended to contribute to.

A graphical representation of these terms and relations is given in Fig. 2.

We note that it is unlikely that the different model consumers each have the exact same goal for which they wish to employ the domain model. Model consumer goals may be overlapping, disjunct, contradictory, or any combination thereof. The modeler therefore is not guaranteed to have a clear goal to work toward when creating the model. Instead, it is their task to size up the expected model consumers, and formulate some *model objective* that, if achieved, can be expected to contribute to the model audience goal set. Once such a model objective is formulated, the modeler can pursue this objective, with the actual domain model the concrete result that (wholly or partly) achieves the model objective. This notion is graphically presented in Fig. 3.

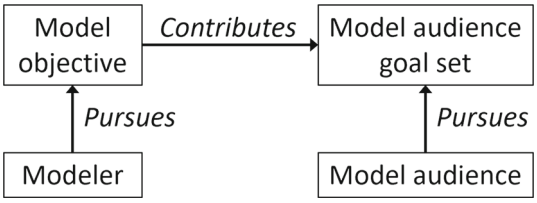

Fig. 3 To have the model contribute to the goals in the model audience goal set, the modeler pursues a model objective

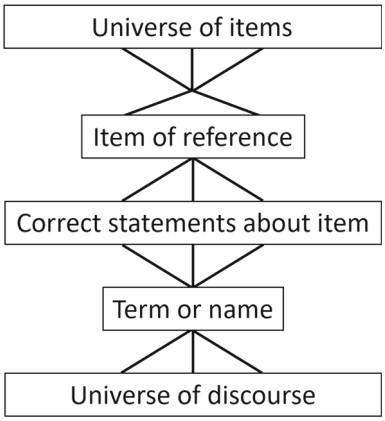

Fig. 4 Concept construction. Reproduced from [10, fig 1]

Given the above, as well as the definitions from Sect. 3.6, we now come to the following definition:

Definition 13 Model Objective: a (usually specific, concrete, and near-term) result that a modeler pursues, as they expect it to contribute to their model audience's goal set.

A domain model has a referent, and therefore captures a concept

We now turn to the essential attribute *domain* from definition 7. This term describes what a domain model is referring to, it is the *referent* of the domain model. Since philosophy of language applies to domain models [18, ch.2], we can employ concept theory [10] to clarify the term *referent*. Concept theory offers a mapping between the referent as an *item* that is positioned in some *universe of items*, the *concept* as a "unit of knowledge", and a *term* as the label we use to denote that concept in some *universe of discourse*. The central idea for concept construction from [10] has been reproduced in Fig. 4.

Concept theory generally connects the item of reference, the referent, to the word we use to refer to it, as well as the ideas we have about it. This connection is labeled in [10] as the concept triangle, and is graphically presented in Fig. 5.

Under concept theory, the referent is an item[6] within a universe of items, which comprises everything from ideas and

[6] Note, however, that the item, the referent, need not be singular. It often (arguably always) concerns some constellation or aggregation of

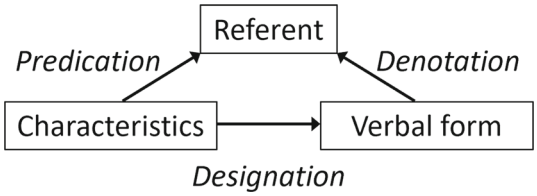

Fig. 5 Concept triangle. Reproduced from [10, fig 3]

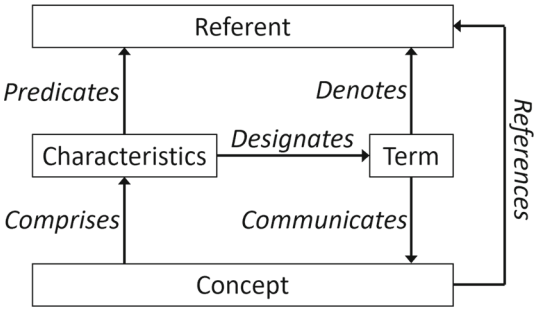

Fig. 6 Relations between referent and concept

objects to actions and dimensions. This referent can then be understood by a human mind as a *concept*, and denoted using a label, name, or *term*. Having a term with which to communicate the concept makes it possible to employ the concept in an (inherently transitive [41]) *universe of discourse* [55]. Under concept theory[7] a concept is itself made up of both a number of *characteristics* and some *term* that denotes that referent, and can be used to communicate the concept as a whole. These properties are represented in Fig. 6.

With this summary of concept theory in place, we can now investigate its application to domain models. We start by noting that a domain model attempts to represent a referent (which we had labeled "domain" in definition 7). That referent is then situated in some specific *universe of items* (concept theory, Fig. 4), which can be the intransitive reality, but also some future, contingent, or otherwise imaginable state of affairs. This implies that a domain model must itself pertain to that specific universe of items, and no other. This universe of items provides the *context* for the referent. The domain model, and all it entails, can be assumed to be valid when discussed in a context that aligns with the specific universe of items, but cannot be assumed to be valid when discussed in another context. To be able to refer to this context, we could employ the term *possible world* from philosophy [40], but this term is usually not readily understood outside of philosophy. For the activity of modeling, we therefore propose the term *Referent universe* as denoting the specific universe of items that the domain model is pertaining to.

To be clear: the "real world" that is recognized by positivists, realists, empiricists, and the like, is a universe of items. Any past or future state of the real world is also a universe of items, distinct from the real world because of its position in time. But the mental image of reality that we have in our minds is *not* a universe of items, but rather a set of concepts that reference it.

We can now craft the following definitions:

Definition 14 Referent: an item positioned in a referent universe.

Definition 15 Referent universe: some universe of items, in which the item that is of interest to the modeler exists.

In both of these definitions, an *item* can be anything: "... a single object, a set of objects considered as a unit, or a property, an action, a dimension, etc. or any combination of these." [10, p.143].

Next, we note how a domain model is expected to contribute to the cognitive goals of a set of model consumers (definitions 9 through 12). In terms of concept theory, this means that the domain model is communicating units of knowledge, i.e., concepts. As philosophy of language applies to domain models, we can say that a domain model serves in a discourse between modeler and audience (even though this discourse will resemble a monologue). This means that the scope that a domain model covers (or intends to cover, or is expected to cover) maps to the *universe of discourse* from concept theory. In the context of modeling, we label a domain model's transient universe of discourse the *model universe*,[8]. Thus we arrive at these two definitions:

Definition 16 Concept: a knowledge unit comprising the characteristics of a referent, a *term* or a name [10, p.144], and a meaning.

Definition 17 Model universe: the universe of discourse within which the model is situated.

We want to stress the importance of properly distinguishing between the referent universe and the model universe. The former describes the circumstances and context of the referent that the domain model is trying to reference, while the latter provides the universe of discourse, containing the context and background for the domain model itself.

To find the relations between the model, referent, and their respective universes, as well as their relation with concepts, we can make use of the semiotic triangle by Ogden and Richards, as we previously did in [52]. Where the semiotic

sub-items, on an arbitrary number of sub-levels. The "domain" from definition 7 then forms the outer wrapper of the constellation or aggregation.

[7] Concept theory is not the only theory offering such an account; see for instance [23], or consider the semiotic triangle [52].

[8] This is not a new insight. For instance in [44, ch.3] the model universe is recognized, although labeled the "physical domain".

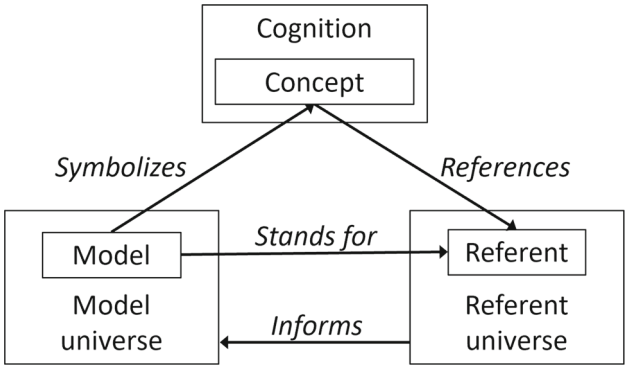

Fig. 7 Model, referent and concept in the semiotic triangle

triangle has at its corners: *symbol*, *thought or reference*, *and* referent, we now position the model, the concept, and the referent, respectively. Connecting these with the three relations from the semiotic triangle (symbolizes, references, and stands for), we arrive at Fig. 7. Additionally, we note that the referent universe *informs* the model universe.

A domain model is a social artifact

Definition 7 declares a domain model to be a specific kind of social artifact, stemming from the observation that domain models are created by humans, and then used in a social setting. Given its position in definition 7, the concept of "social artifact" is a superordinate concept for "domain model". Therefore a more explicit definition than the terse description we gave in [51] is in order. Based on dictionary definitions such as [6] we state:

Definition 18 Social artifact: something produced by huma ns, to be employed in a social setting.

A domain model is cost-effective, and therefore has associated value and cost

We are interested in the costs and benefits that domain models bring to their users. These costs and benefits can be evaluated using the concept of Return on Modeling Effort (RoME) [51]. But neither the costs nor the benefits of creating and using domain models are reflected in definition 7 as we have been using up to this point in time. By contrast, some of the extant definitions for models in general do take this into account, to a certain extent.

Rothenberg recognized that any model is characterized, among other aspects, by it being *cost-effective*, stating that "It is more cost-effective to use the model [...] than to use the referent itself" [56]. Given this, we argue that cost-effectiveness must be an essential characteristic of a domain model, and thus must be added to our definition for it.

Adding cost-effectiveness to definition 7 requires also obtaining a definition for this characteristic itself, per the reasoning in Sect. 3.2. But for this, Rothenberg offers little assistence: cost-effectiveness is simply considered the difference between the cost of manipulating the model and the cost of manipulating the referent directly. But this approach does not consider the possibility of new or extra value arising from the *creation* of the model itself, nor are the costs that are associated with the creation of the model taken into account.

A slightly more elaborate appraisal of the value of models is provided by [45], which considers the use of models in the domain of decision analysis. On this approach, some actor can either take action without making use of models, or they can have a model created, and then take action supported by the information provided by the model. In [45], the difference in the value of the actions taken without and with the model is considered the value that modeling brings, expressed as the Expected Value of Modeling (EVM). This approach focuses on value rather than cost; it does not take into consideration the costs that are associated with the creation and use of the model.

Inspired by the work in [45], we now propose the following expression for cost-effectiveness:

Definition 19 Model cost-effectiveness: the ratio between the sum total value that the model audience can extract from the model, and the sum total value of the effort of constructing the model and of consuming the model.

In this definition, the two essential attributes are:

- **The sum total value that the model audience can extract from using the model**: this is the sum of the value that befalls each of the model consumers when they are using the model, minus the value that would befall them if they did not use the model (say, if they used the referent directly, or did nothing at all);
- **The sum total value of the effort of constructing the model and of consuming the model**: constructing the model will have associated cost; consuming the model may also have associated cost, such as the time spent on it. The sum of these costs will have a non-zero value.

Both the value that the model audience obtains, and the cost that the modeler and model audience expend, need to be in the same currency; for example money, time, transactions, or person-hours. Furthermore, you cannot choose a currency for which the cost of creating and consuming the model is zero. If both these conditions are satisfied, then the cost-effectiveness ratio is a dimensionless number. If this number is higher than one, then the use of the model is cost-effective (for the chosen currency).

4.3 An expanded definition for "domain model"

Given the considerations and definitions from Sect. 4.2, we hereby propose the following definition for *domain model* as an expanded form of definition 7:

Definition 20 Domain model: a social artifact reflecting a concept, intended to cost-effectively pursue a particular cognitive objective in support of a particular audience.

The definitions for the essential characteristics in this definition are given here:

- Social artifact: definition 18;
- Concept: definition 16;
- (Model) cost-effectiveness: definition 19;
- (Model) objective: definition 13;
- (Model) audience: definition 11.

4.4 A definition for "enterprise architecture model"

It should be noted that every term and relation in the remainder of this paper refers specifically to the use of modeling within the context of enterprise architecture. But in creating the labels we will abbreviate the prefixes "enterprise architecture" to EA, and "enterprise architecture model" to EAM, to shorten the labels and improve readability. Thus we write *EA modeler* instead of *enterprise architecture modeler*, *EAM* instead of *enterprise architecture model*, *EAM consumer* instead of *enterprise architecture model consumer*, et cetera.

As stated in Sect. 1, our interest is in EAMs and their quality. A definition for EAM can be arrived at by stating the definition of its superordinate concept, followed by the delimiting characteristics [29, 6.2]. We thus start from definition 20, and take into consideration the following points.

The *audience* of an EAM will consist of actors that have an interest in some change within the enterprise. These actors are considered *stakeholders* for the change under consideration. However, not every stakeholder will have the competencies to fully make use of a given EAM, for instance they may not know the used EA modeling language. This person will then still be a stakeholder, but cannot be a member of the EAM audience. (Conversely, if this person must be a member of the EAM audience, then the modeler cannot use that particular EA modeling language for the EAM.) Given these restrictions, we come to the following definition 21 for an EAM consumer.

Definition 21 Enterprise Architecture Model consumer: an actor who is expected to consume a specific enterprise architecture model, on the belief that this will contribute to some cognitive goal of theirs.

In this definition, *consume* entails "without assistance from any other person" (definition 8), and this includes the modeler.

The *EAM audience* as a whole consists of one or more EAM consumers, which leads to the following definition 22:

Definition 22 Enterprise Architecture Model Audience: a set of enterprise architecture model consumers that are each expected to consume a specific enterprise architecture model.

The *cognitive objective* that the EAM intends to pursue is focused on the information needs that the members of this EAM audience have.

The *concept* that the EAM reflects will be the *architecture* under consideration. This term has many definitions, and we will not spend time on it in this paper. Instead we will here adopt the definition from ISO42010-2022 [28, ch.3.2], as this is considered an authoritative source in the profession of enterprise architecture:

Definition 23 Enterprise Architecture: fundamental concepts or properties of an entity in its environment and governing principles for the realization and evolution of this entity and its related life cycle processes.

Applying all of the above to definition 20, we arrive at the following definition 24 for an EAM:

Definition 24 Enterprise Architecture Model: a social artifact reflecting an enterprise architecture, intended to cost-effectively pursue a particular cognitive objective in support of a particular audience.

The definitions for the essential characteristics in this definition are given here (leaving out the prefix EAM):

- Social artifact: definition 18;
- Architecture: definition 23;
- Cost-effectiveness: definition 19;
- Objective: definition 4;
- Audience: definition 22.

Note that the definition for an EAM as presented in definition 24 does not require a model to have a specific *form*. It does not require an EAM to be graphical (expressed neither in informal boxes-and-lines, nor in a formal graphical modeling language such as ArchiMate or UML). *Any* artifact can serve as an EAM: a running text, a presentation slide deck, an animation or movie, a set of mathematical equations, a physical maquette, a digital simulation, or even something as ephemeral as a role-playing game or an interpretive dance.[9]

[9] An interpretive dance being an EAM is an extension to the extreme. We are not aware of any enterprise architecture ever communicated in this way. However, on principle this is not an impossibility.

Also note that on the given definition an EAM need not have a *singular* form. The social artifact that serves as the EAM can be the sum of different forms, such as a running text with diagrams and mathematical equations.

5 Domain lexicon

Making good use of the definitions obtained in the previous Sect. 4, we will now compile a lexicon for the domain of enterprise architecture modeling. This lexicon consists of all the terms and their definitions for the domain, augmented with the relations between these concepts, and captured in OntoUML.

5.1 Positioning the architecture

We begin by considering the position of the EAM with respect to the system that it intends to capture, and the position of both in their respective universes. For this, we look at Sect. 4, specifically the concepts of referent, referent universe, model, and model universe, as captured in Fig. 7. These concepts then lead to EA-specific versions, to begin with the referent and its universe.

In EA, the referent equates to the part of the enterprise that is considered for change. This may be anything, from a value chain down to a sub-process, from an entire IT infrastructure landscape to a single system, from an information landscape to a single data set, and more. We choose for this referent the label "system"; consequentially the universe of items in which such a system is positioned can best be labeled the "system universe". Adapting definitions 14 and 15, we arrive at the following, EA-specific, definitions:

Definition 25 System: an item of interest within the enterprise that is the subject of an enterprise architecture model, whereby "item" can be anything from ideas and objects to processes and information, or aggregations thereof.

Definition 26 System universe: the "universe of items" that harbors the system that is the subject of the enterprise architecture model; the context in which that system exists.

The EAM that has the system as its subject has itself been defined in definition 24. But as the previous section, and specifically Fig. 7 indicate: an EAM exists within a specific universe that contains and provides the context for that EAM. This is captured in the following definition 27:

Definition 27 Enterprise Architecture Model universe: the "universe of discourse" that covers the part of the enterprise and its surroundings that fully envelopes the system that is the subject of the enterprise architecture model, including all parts of the enterprise that themselves change if the system is changed.

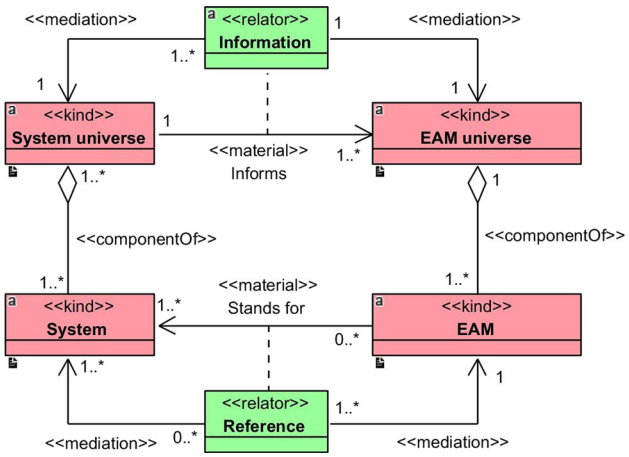

Fig. 8 Positioning the EA model

The four terms above are related by four relations. Two of these are straightforward: an EAM is part of an EAM universe, and a system is part of a system universe. The third relation is the one between EAM and the system that it models; from the semiotic triangle and Fig. 7 we see that an EAM *stands for* the system that is being modeled. The fourth relation is between the two universes: we say that the system universe *informs* the EAM universe, in that information present in that EAM universe must come from, and correspond with, the system universe.

The representation in OntoUML of these four terms and their interrelations is provided in Fig. 8. Note that the two material relations are associated with a ⟨⟨relator⟩⟩ concept, which disambiguates the relation, providing the material conditions that act as truthmakers for the relation [16]. We label the relators and include them in the OntoUML diagrams, but we will not document them in this paper, instead relegating that work to future publications.

5.2 Conceiving the architecture

We now turn to the question how the EA that is to be captured in the EAM gets "conceived". This conception is carried out by the *EA modeler*, a role assigned to an actor, that entails working on and with EAMs.

Definition 28 Enterprise architecture modeler: a role, assigned to an actor, requiring the creation, maintenance, or expansion of an enterprise architecture model.

In the first phase of modeling, the EA modeler will need to create in their mind a conception of the system that is under consideration, existing in some system universe—as defined in Sect. 5.1. In order to be able to consider the system itself, the modeler inevitably will also have to consider the context of the system, and thus the relevant sections of the system universe; this means the modeler needs to *conceive*

both the system itself, and at a minimum that part of the system universe with which the system interacts, or which has a (direct or indirect) influence on the system. Note that in some cases, this system universe may actually coincide with current reality (the *world*); this pertains in particular when the EA modeler is creating a baseline architecture, documenting what is already there. In that case certain elements[10] of that system can likely be conceived by directly *perceiving* the world.

Definition 29 World: the state of things as they are, outside and independent of any observer.

Once the concept of the system manifests itself in the mind of the EA modeler, they can then work with that concept, in order to investigate the *enterprise architecture* of that system. This will lead to the forming in their mind of another concept, namely the *architecture concept*. This concept comprises the knowledge about the architecture of the system that the EA modeler will (attempt to) capture in the EAM.

Definition 30 Architecture concept: a knowledge unit (characteristics, meaning, and a term or label) that comprises the enterprise architecture of a system.

The representation in OntoUML of the terms and relations as presented above is depicted in Fig. 9.

5.3 Modeling the architecture

In the second phase of modeling, the EA modeler consults the architecture concept that they conceived in their mind. From that, they can now construct the actual *EAM*, as defined in definition 24.

We now look closer into the EAM itself, as a thing that the EA modeler is constructing. It must have some sort of perceptible form (for example a drawing on paper, or an electronic image), or it could not be read by its intended audience. Furthermore, it must have an information content, or it could not serve the cognitive goals of its intended audience. The distinction between form and content is discussed in [38], which states that object M carries cargo χ, and that this object M corresponds to model μ (explicitly using the word "is" for that correspondence relation). We introduce the following terms to refer to object M and cargo χ:

- an EAM has an *EAM content*, the information that reflects the architecture that the EA modeler attempts to capture.
- The EAM content is carried by some form of *EAM embodiment*.

[10] Note, however, that some elements of the system are not directly perceivable even in the real world. As an example, consider some law or regulation that exist in the real world. EA modelers will not be able to perceive this law or regulation itself. Instead they can only read the text that records that law or regulation, and conceive of it in their mind.

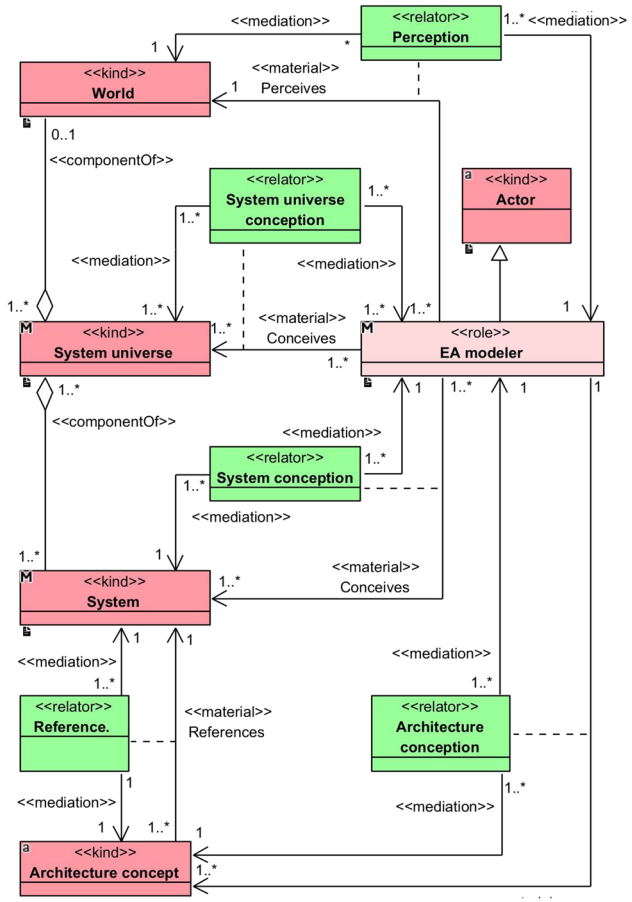

Fig. 9 How the modeler conceives an enterprise architecture

These terms have the following definitions:

Definition 31 Enterprise Architecture Model Content: the information that reflects the enterprise architecture that an enterprise architecture modeler attempts to capture.

Definition 32 Enterprise Architecture Model Embodiment: the perceptible form of an enterprise architecture model that carries the enterprise architecture model content, and in doing so expresses the architecture concept.

The four material relations that connect the terms above are:

- Between EA modeler and architecture: the EA modeler holds in his mind the architecture as a concept, a unit of knowledge. When creating the EAM, he will be *consulting* this knowledge.
- Between EA modeler and EAM: this relation covers the fact that EAMs are created by EA modelers; a common term for that is *constructing*.[11]

[11] Note that the term *modeling* is less suitable, since that is usually understood to include the conceptualization of the architecture, not "just" the creation of what will become the EAM.

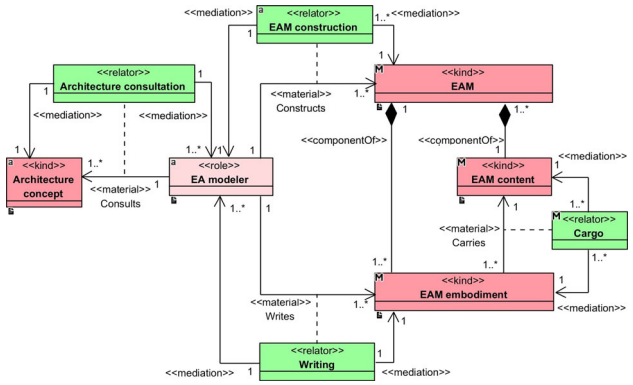

Fig. 10 The terms surrounding the modeling of the EA

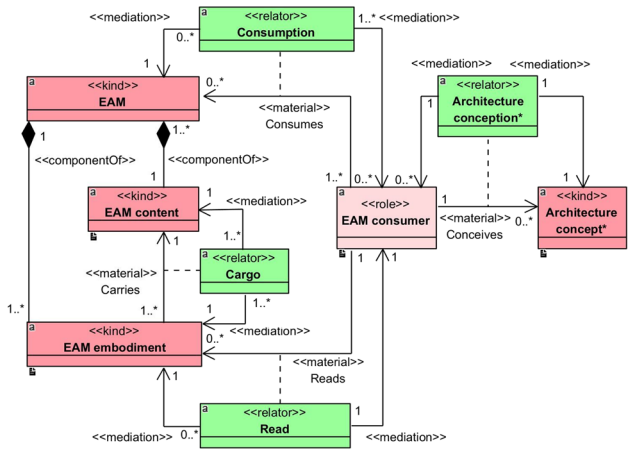

Fig. 11 OntoUML representation of how an EAM consumer consumes an EAM

- Between the EA modeler and the EAM embodiment: the actual activity that produces the embodiment is dubbed *writing*, reflecting the fact that an EAM usually is embodied in diagrams and texts.
- The relation between EAM embodiment and EAM content is dubbed by [38] to be *carrying*. We also will employ this label.

There are two more relations, but these are not material. According to [38] a model embodiment *is* a model, or conversely that a model *has* a model embodiment. But as we wish to avoid the verb *is*, we select *composes*. Using this relation, an EAM is composed of both an EAM embodiment, and EAM content.

Note that no direct relation is given between the EA modeler and the EAM content. The EAM content is carried by the EAM embodiment, and is not directly accessible by the EA modeler, other than by manipulating that EAM embodiment.

The representation of the terms and relations in OntoUML is given in Fig. 10.

5.4 Consuming the enterprise architecture model

There is at least one *EAM consumer*, someone who is potentially or actually going to consume the model—see definition 21. Such an EAM consumer, after consuming the EAM, will have in their mind as a concept some part of the architecture that the EA modeler used to construct the model. However, the concept conceived by the EAM consumer likely is not exactly the same as the concept that the EA modeler conceived, as communication is inherently lossy, and no EAM consumer will have exactly the same background knowledge that the EA modeler anticipated. Furthermore, the concept that the EAM consumer has in their mind may only cover a part of the architecture, as the EAM consumer may not have (needed to) consume the EAM in its entirety. As a result, the concept in the mind of the EAM consumer diverges from the architecture in the mind of the EA modeler, both in scope and accuracy. It is therefore a different concept, and consequentially gets its own label, namely *architecture concept**. Derived from definition 30, this concept gets the following definition:

Definition 33 Architecture concept*: a knowledge unit that comprises the enterprise architecture of a system, or part thereof, as it is learned by a stakeholder from an enterprise architecture model.

When an EAM consumer consumes an EAM, there are three relations in play:

- Between the EAM consumer and the EAM: this relation is named *consuming*, referencing definition 8.
- Between the EAM consumer and the EAM embodiment: this term is dubbed *reading*, mirroring the writing relation assigned in Sect. 5.3.
- Between the EAM consumer and the architecture concept*: by consuming the EAM, the consumer will learn about the content of the EAM, constructing some mental model that matches to a certain extent the content of the EAM. This construction of a mental model is labeled *conceiving*; but the ⟨⟨relator⟩⟩ for this relation is a different one from the ⟨⟨relator⟩⟩ in Fig. 9. As it pertains to the architecture*, it will have as ⟨⟨relator⟩⟩ *Architecture conception**

The representation in OntoUML of the above is provided in Fig. 11.

5.5 The audience for an enterprise architecture model

For EAM audience, we previously arrived at definition 22, for EAM consumer at definition 21. For the goals of individual

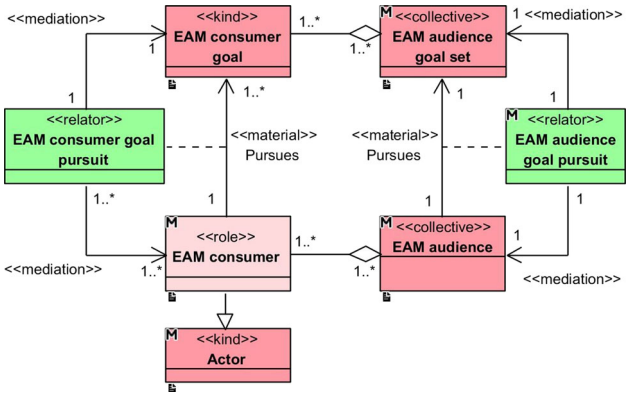

Fig. 12 OntoUML representation of the audience for a given EAM

EAM consumers, and for the EAM audience as a whole, we can specialize the definitions 10 and 12. We then obtain:

Definition 34 Enterprise Architecture Model consumer goal: a cognitive goal, pursued by an enterprise architecture model consumer, connected with the enterprise architecture that the enterprise architecture model attempts to capture.

Definition 35 Enterprise Architecture Model audience goal set: the aggregated set of enterprise architecture model consumer goals that an enterprise architecture model is intended to contribute to.

In line with the relations from Fig. 2, the terms above are related using:

- two material relations "pursues" between consumer and their goal, and between audience and its goal set;
- two aggregation relations, between goal and goal set, as well as between consumer and audience.

The representation in OntoUML of these specialized terms and relations is provided in Fig. 12.

5.6 The relations between EAM, architecture concept, and architecture concept*

The architecture concept* will correspond to a greater or lesser extent to the architecture concept as conceived by the EA modeler. In an ideal situation (perfect EA modeler, EAM, and EAM consumer) the architecture concept* will correspond fully to the architecture concept. But if the quality of the EAM is less than perfect, or if other factors intervene, then the correspondence will be less than complete. This relation thus plays a major role in EAM quality, such that it warrants a formal definition:

Definition 36 Correspondence: the degree to which the interpretation of an enterprise architecture model by an enterprise architecture consumer agrees with the architecture concept as conceived by the enterprise architecture modeler.

Since we can understand the EAM in three different ways (as an EAM embodiment, as EAM content, and as the whole of the EAM), there are three ways to relate the EAM to the architecture:

- *Captures*: prior to this section, we have already employed the term *capturing* for the relation between architecture concept and EAM.
- *Reflects*: the EAM content is the information carried by the EAM, it is not the architecture concept itself. A word that expresses this epistemological ambivalence is *reflecting*. The better the EAM content is reflecting the architecture concept, the better the EAM is from an information perspective.
- *Expresses*: we used the relation *writing* between EA modeler and EAM embodiment. Since writing is a form of expression, for the reverse relation we state that the EAM embodiment *expresses* the architecture.

When an EAM consumer is consuming an EAM and conceiving the architecture concept*, they are doing the reverse that the EA modeler was doing. It is therefore convenient to have the terms for the relations between architecture concept* and EAM/EAM content/EAM embodiment mirror the terms used between those and the architecture concept:

- *Interprets*: the opposite side of expressing;
- *Matches*: this term is selected to bridge the epistemic uncertainty between architecture concept* (a mental construct) and EAM content (information carried by the embodiment).
- *Is understanding*: When an EAM consumer is consuming the EAM, they are attempting to understand it. As a result of this activity, the architecture concept* is the understanding that the EAM consumer gets from the EAM.

The OntoUML representation of these seven material relations is provided in Fig. 13.

5.7 How the EA model satisfies the EAM audience

The preceding texts have already discussed how the EA modeler is constructing the EAM so that the members of an EAM audience can consume it. What has not been covered is what the EA modeler needs to do to satisfy that EAM audience.

In Sect. 5.3 we formulated that in general an *EAM audience* has an *EAM audience goal set*, a set of goals from the individual EAM consumers that the EAM is required to contribute to. The goals in this EAM audience goal set may all converge entirely, may be entirely spread out and non-overlapping, or anything in between. Regardless, the EA modeler needs to create an EAM that maximally contributes to the EAM audience goal set as a whole. To this end, the

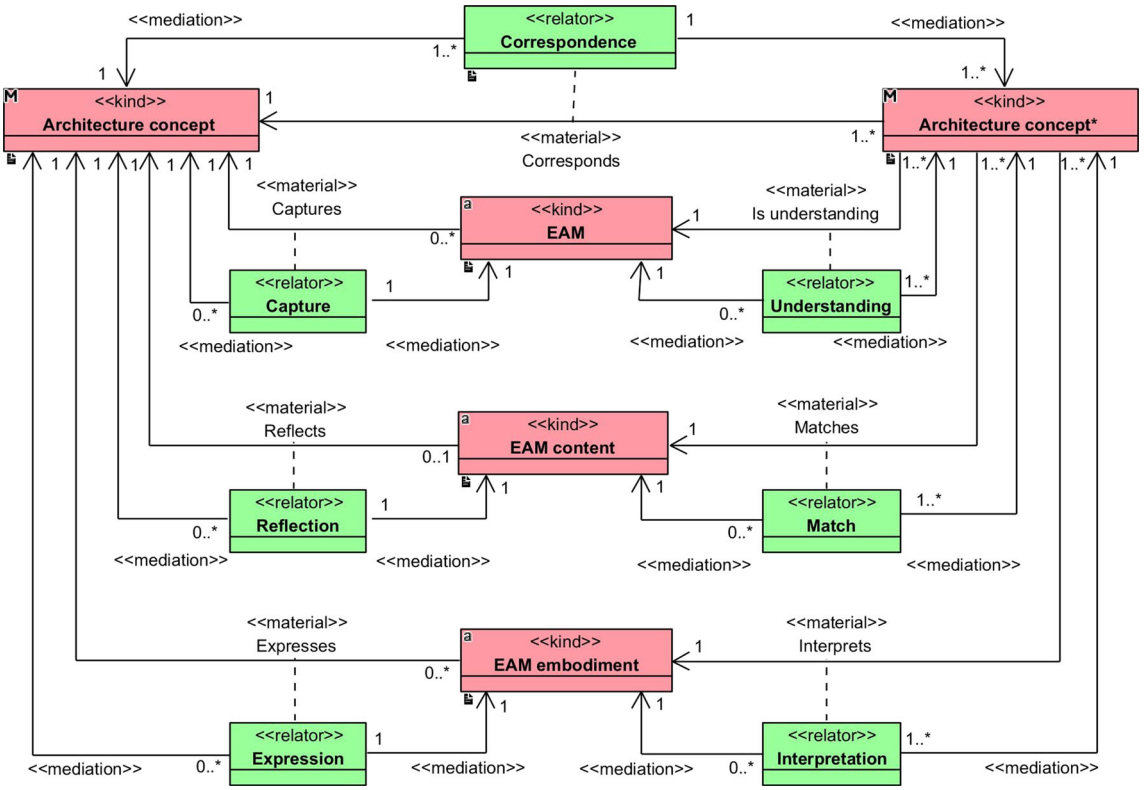

Fig. 13 OntoUML representation of how the architecture concept* corresponds with the architecture concept

modeler will need to formulate for themself an *EAM objective*, which can inform the construction of the EAM in such a way that the resulting EAM will maximally contribute to the EAM audience goal set. Put in a definition:

Definition 37 Enterprise Architecture Model Objective: the objective that an enterprise architecture modeler needs to meet, in order for the enterprise architecture model to provide the maximum contribution to the goals of the enterprise architecture model audience.

Having the EAM objective in place gives rise to the following new relations:

- As Sect. 3.6 shows objectives to contribute to goals, so does the EAM objective *contributes* to the EAM audience goal set;
- Having formulated an EAM objective, the EA modeler can decide what to include in the EAM, and how. We say that the EAM objective *informs* the EAM;
- To this end, the EA modeler needs to *pursue* the EAM objective while creating the EAM;
- As Sect. 3.6 shows results to contribute to goals, and the EAM is a result, it is fair to say that the EAM itself *contributes* to the EAM audience goal set as well;

An OntoUML representation of the terms and relations discussed above (as well as some discussed prior) is given in Fig. 14.

5.8 How an EAM consumer judges an EAM's quality

The quality of a product or service is determined by the perception of the consumers of that product or service [31, ch.1].[12] For the product that is the EAM, its quality is determined by the perception of its consumers: the members of the EAM audience. To capture how the EAM audience as a whole comes to a quality judgment for the EAM that they are consuming, we start from an individual EAM consumer. That actor is pursuing their own particular EAM consumer goal; it is to that end that the EAM consumer will consume the EAM. This then leads to the EAM consumer conceiving in their mind an architecture concept*. The EAM consumer can then engage in *consulting* this conception.

After making use of the EAM, the EAM consumer will be *forming* an *EAM quality judgment* based on at least the following criteria:

[12] Differing views of quality exist. The view espoused in [31] broadly follows the approach from Total Quality Management as espoused by Deming, Juran, Ishikawa, and more—see [66].

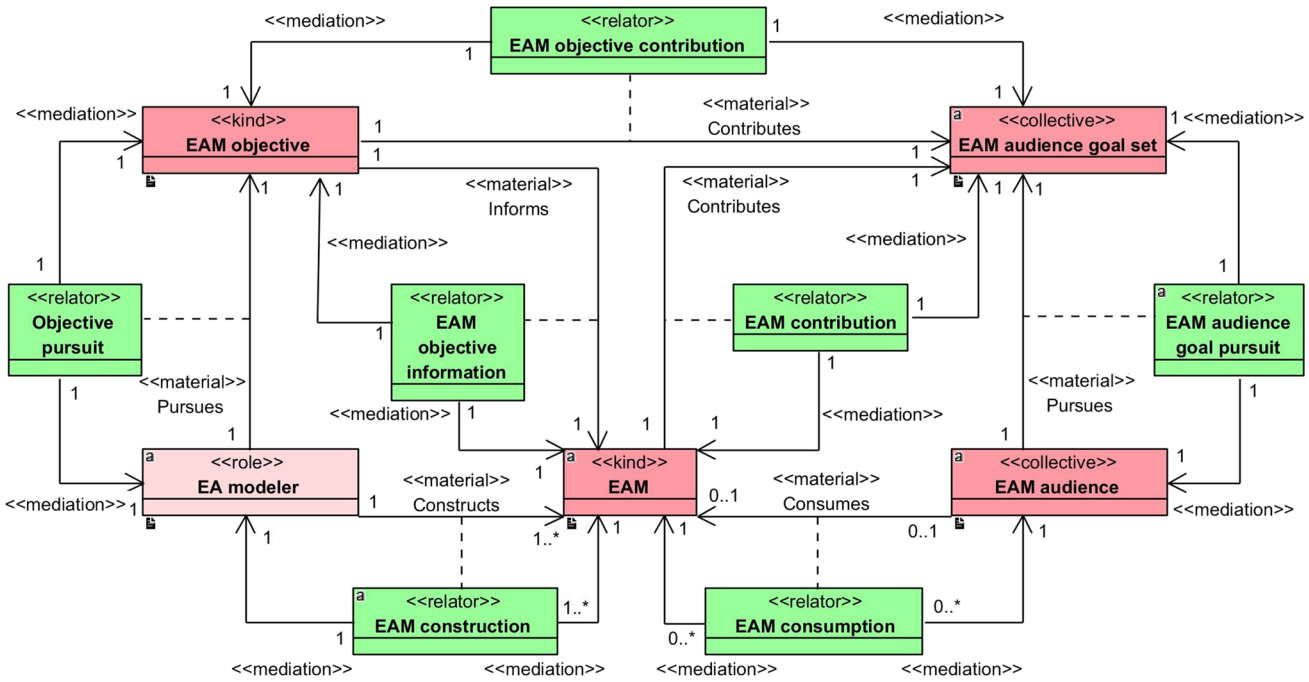

Fig. 14 OntoUML representation of how an EA modeler is pursuing an EAM objective

- The completeness and the quality of the information that they extracted from the model;
- The ease with which this architecture concept* can be integrated in this person's mind;
- the extent to which this person's understanding of the architecture concept* actually contributes to their goals.

When an EAM consumer is *judging* how well the consumption of the EAM has served them, then they are judging the EAM itself. This consideration now provides the definition for *EAM quality judgment*:

Definition 38 Enterprise Architecture Model quality judgment: the judgment by an enterprise architecture model consumer of the ease with which they can employ the information in the enterprise architecture model in the pursuit of their goals, and the extent to which their understanding of this information actually contributes to those goals.

The graphical representation of the terms and relations discussed above is given in Fig. 15. Note that in OntoUML terms the EAM quality judgment is not a ⟨⟨kind⟩⟩, but a ⟨⟨relator⟩⟩, signaling that it is not a functional complex, but it is the truthmaker of the material relation *judges*.

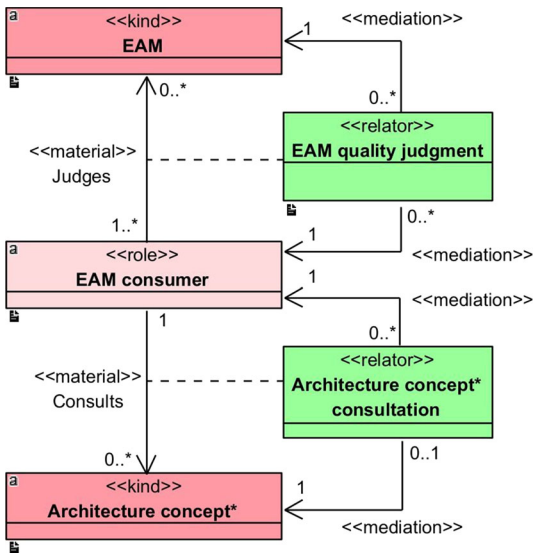

Fig. 15 OntoUML representation of how an EAM consumer arrives at an EAM quality judgment

5.9 A model for EAM quality

When describing the quality of an *EAM*, the first term that is employed is always *model quality* as a quality, characteristic, or feature that the EAM is *having*. To obtain an understanding for what this actually entails, we turn to the ISO-25012 standard [26] that offers a data quality model. This is fitting, since an EAM is essentially a collection of data, describing the enterprise architecture.

The conceptual model offered by [26, ch.5] in the form of the "data quality model" is straightforward: quality is captured using fifteen distinct "quality characteristics". A slightly more generic conceptual model is offered by [31], which holds that any consumer that obtains a product or service "is essentially buying a bundle of attributes" [31, p.2].

Fig. 16 OntoUML representation of the conceptual model for EAM quality

Noting that "characteristic" and "attribute" are used synonymously by [26] and [31], we hold that *EAM quality* is an aggregation of *EAM quality attributes*. Being an aggregation, the relation between the two would be *having*.

Given the above, a straightforward conceptual model for the quality of an EAM will contain the following two concepts:

Definition 39 Enterprise Architecture Model quality attribute: a characteristic of an enterprise architecture model that contributes to an enterprise architecture model quality judgment.

Definition 40 Enterprise Architecture Model quality: the aggregation of all enterprise architecture model quality attributes.

The OntoUML representation of this model is provided in Fig. 16.

There currently is no consensus on the contents of the set of attributes that comprise EAM quality. The set of 27 attributes compiled from the literature in [59] may serve as a beginning for a more systematic investigation, but we will not initiate such an investigation in this paper.

6 Validity

6.1 Ontological analysis: capturing the ontology in OntoUML

The first step in validating the ontology for EAMs is performing an ontological analysis, by encoding the ontology in OntoUML. For this, we have entered the ontology in a specialized modeling tool.[13] This encoding allows for two specific tests:

- A successful mapping of the terms onto OntoUML classes shows that the concepts identified in the ontology have a sound ontological foundation, as described in UFO [18].
- The plug-in provides a syntactical analysis that can identify syntactical issues with the OntoUML model.

Our ontology passes both these tests.

[13] The tooling used is Visual Paradigm 17.1 Community edition, with the OntoUML 0.5.3 plug-in.

6.2 Using the OntOlogy Pitfall Scanner! (OOPS!)

For this we make use of the OOPS! catalog [73]. Of the forty-one pitfalls listed, sixteen pertain to the ontology in the form published in this paper. The other pitfalls deal specifically with ontologies that contain properties, are encoded in OWL, are published online, or are part of a linked data effort. As our ontology does not have these characteristics, these other pitfalls do not pertain.

In the list below, each relevant pitfall is listed along with a terse description, followed by the measures we took to avoid it:

- P01. Creating polysemous elements: An ontology element (class, object property or datatype property) whose identifier has different senses is included in the ontology to represent more than one conceptual idea or property; and P03. Creating the relationship "is" instead of using "rdfs:subClassOf", "rdf:type" or "owl:sameAs"; and P07. Merging different concepts in the same class: A class whose name refers to two or more different concepts is created. We formulated our naming conventions (Sect. 3.3) to explicitly avoid homonymy and polysemy, including common and generic verbs. We did not use *is*, or conjugations of it, at all.
- P02. Creating synonyms as classes: Several classes whose identifiers are synonyms are created and defined as equivalent (owl:equivalentClass) in the same namespace. Since we methodically progressed through the definition of each class in table 2, we have no classes that are equivalent.
- P04. Creating unconnected ontology elements: Ontology elements (classes, object properties and datatype properties) are created isolated, with no relation to the rest of the ontology. Our ontology is fully captured in the nine Figs. 8, 9, 10, 11, 12, 13, 14, 15, and 16. All of these are connected via the ⟨⟨kind⟩⟩ *EAM*, except for 9 and 12, but the former is connected to 8 via *System*, and the latter to 11 via *EAM consumer*. Thus, all elements are interrelated into a single group.
- P05. Defining wrong inverse relationships: Two relationships are defined as inverse relations when they are not necessarily inverse; and P06. Including cycles in a class hierarchy: A cycle appears when some class A has a subclass (directly or indirectly) B, and at the same time B is a superclass (directly or indirectly) of A; and P17. Overspecializing a hierarchy: The hierarchy in the ontology is specialized in such a way that the final leaves are defined as classes and these classes will not have instances; and P21. Using a miscellaneous class: This refers to the creation of a class with the only goal of classifying the instances that do not belong to any of its sibling classes (classes with which the miscellaneous problematic class

shares a common direct ancestor). We have no inverse relations in the ontology, nor subclasses, nor hierarchies, nor miscellaneous classes.
- P09. Missing domain information: Part of the information needed for modeling the intended domain is not included in the ontology. Our analysis has put down a foundation that is itself complete. We do recognize that this foundation is for a domain of limited scope. For instance, we have not yet included modeling with multiple EA modelers.
- P10. Missing disjointness: The ontology lacks disjoint axioms between classes or between properties that should be defined as disjoint. Our terms do not include sets of disjoint terms, so disjoint axioms are not required.
- P22. Using different naming conventions in the ontology: The ontology elements are not named following the same convention. A single naming convention has been provided in Sect. 3.3, and followed throughout this paper.
- P24. Using recursive definitions: An ontology element (a class, an object property or a datatype property) is used in its own definition. Every definition has been created using the method described in 3.2, specifically the criteria for definitions. Recursion, whereby the definiendum appears (directly or indirectly) in the definiendum, does not occur.
- P25. Defining a relationship as inverse to itself: The ontology does not have symmetric properties that could cause this pitfall.
- P32. Several classes with the same label: Two or more classes have the same content for natural language annotations for naming. Table 2 shows that every term (label) appears only once.

This analysis shows that we have avoided the common problems ("pitfalls") that may occur in ontology development.

Table 2 Domain lexicon for enterprise architecture modeling: twenty terms, their definitions, and their OntoUML type

Lexical term	Definition	Type
Actor	5	Kind
Architecture concept	30	Kind
Architecture concept*	33	Kind
Correspondence	36	Relator
EA modeler	28	Role
EAM	24	Kind
EAM audience	22	Collective
EAM audience goal set	35	Collective
EAM consumer	21	Role
EAM consumer goal	34	Kind
EAM content	31	Kind
EAM embodiment	32	Kind
EAM objective	37	Kind
EAM quality	40	Quality
EAM quality attribute	39	Attribute
EAM quality judgment	38	Relator
EAM universe	27	Kind
System	25	Kind
System universe	26	Kind
World	29	Kind

7 Results and discussion

7.1 Results

We methodically investigated enterprise architecture modeling, making use of solid theoretical foundations. This has yielded twenty domain-specific terms, each with a formal definition. These terms have been compiled in table 2.

Between the concepts that are denoted by the uncovered terms, there are forty-one domain-specific relations, which have been named and described. For two of these relations, a formal definition has been drafted, although the relationships have not been documented in-depth. The relations are summarily listed in table 3.

The ontology as a whole has undergone multiple checks to ensure content validity, and passed all of those. Application validity, whereby the ontology is put in practice to see to what degree it serves its intended purpose, has not yet been tested.

On this ground it is justified to state that the ontology described goes a long way toward answering the question "what constitutes a suitable ontology for investigating enterprise architecture model quality?" that we posed in Sect. 1.

7.2 The value of EAMOn

Value for researchers

As discussed in Sect. 1, currently little scientific research is available that addresses the theoretical underpinnings for EA modeling (with the exception of visualization of models). The results of this paper form a first step toward a theory of EA models, and a well-founded, suitable ontology with which to capture and describe ideas surrounding EA models and EA modeling. As such, the ontology can jump-start new research into EA models and EA model quality. Furthermore, when different researchers each use EAMOn as their ontological framework, their results become more compatible, further stimulating and propagating the research. Finally, we believe the ontology and underlying theories can readily be adapted to suit more generalized types of models.

Table 3 Domain lexicon for enterprise architecture modeling: forty-one relations

Relation	Definition	From/to
Aggregates		EAM Universe → EAM; EAM audience → EAM consumer
		EAM audience goal set → EAM consumer goal; system universe → system
		System Universe → world
Captures		EAM → architecture concept
Carries		EAM embodiment → EAM content
Composes		EAM → EAM content; EAM → EAM embodiment
Conceives		EA modeler → system; EA Modeler → system universe
		EA Modeler → architecture concept; EAM consumer → architecture concept*
Constructs		EA modeler → EAM
Consults		EA Modeler → architecture concept; EAM consumer → architecture concept*
Consumes	8	EAM consumer → EAM
Contains		EAM universe → architecture concept; system universe → system
		System universe → world
Contributes		EAM → EAM audience goal set; EAM objective → EAM audience goal set
Corresponds	36	Architecture concept* → architecture concept
Expresses		EAM embodiment → architecture concept
Has		EAM → EAM Quality; EAM quality → EAM quality attribute
Informs		EAM objective → EAM; System universe → EAM universe
Interprets		Architecture concept* → EAM embodiment
Is understanding		Architecture concept* → EAM
Judges		EAM consumer → EAM
Matches		Architecture concept* → EAM content
Perceives		EA modeler → world
Pursues		EA modeler → EAM objective; EAM audience → EAM audience goal set
		EAM consumer → EAM consumer goal
Reads		EAM consumer → EAM embodiment
References		Architecture concept → system
Reflects		EAM content → architecture concept
Stands for		EAM → system
Writes		EA modeler → EAM embodiment

Value for EA practitioners

While the direct motivation for writing this paper comes from a desire to conduct scientific research into EA model quality, the results are also directly applicable to the activities of EA modelers in practice. EA modelers currently obtain little guidance (if at all) on how to bridge the gap between their subject expertise (EA) and the business that requires this expertise for decision making on change in the enterprise.

Using the ideas captured in the concepts and relations of EAMOn, an EA practitioner can:

- make informed decisions on scoping their model audience;
- based on an EAM objective, obtain clarity on which parts of the architecture of the system under consideration to include or exclude;
- select means by which to represent their model (including what modeling language(s) to use), suitable to the intended audience;
- and investigate selectively to what extent the individual members of the model audience are served by their model.

While the paper in its current form is more elaborated and simultaneously less detailed than EA practitioners generally will appreciate, it is already usable to support the activities described above. But it is relatively easy to work this paper into a form that is more accessible for practitioners. We strive to provide such forms in the near future to the professional community of EA practitioners.

7.3 Critical evaluation

The ontology only accounts for a modeler operating in isolation

While the ontology accounts for differing model consumers, it does currently not deal with a situation in which multiple modelers work on a single EAM. In that situation, could the modelers have a shared, identical conception of the architecture? A single *architecture concept*? Probably not - but what does that mean for the creation of an EAM? This is not covered in this paper; it will be a topic for follow-up research.

A specific version of Concept Theory has been employed

A criticism that can be raised against this ontology is that it leans on a specific version of concept theory [10]. Other versions exist that align with different epistemic positions [24]. Employing a different version may well lead to a different ontology.

This is a valid point. EAMs deal with knowledge (about the EA), and thus any theory on EAMs needs to take epistemology into consideration. We believe that our critical realism aligns well with the Concept Theory proffered by [10], but further work is needed both to strengthen that position, and to provide alternatives that align with a different epistemic stance. The former will be part of our ongoing work; we hope the latter will be taken on by colleagues in the field.

A positivist's appreciation of the transfer of architecture

As stated in Sect. 3.1, our interpretation of the transfer of architecture from modeler to a model consumer via an EAM is built upon the epistemic stance of critical realism. A positivist could argue that the capture of knowledge in the form of the *architecture* concept in the EA modeler's mind can be perfect, and that the transfer to an EAM consumer could therefore conceivably also be perfect, negating any difference between *architecture concept* and *architecture concept** (excluding scope).

It is our position that the transfer of the architecture concept from the EA modeler to an EAM consumer can be imperfect even on positivism, because the EAM itself can be flawed. A positivist must accept this, because the alternative would be to say that a flawed model could still transfer an idea without loss of information; pushed to its extreme, that would indicate that no amount of corruption could interfere with the transfer of information about the architecture concept, which is an untenable position.

7.4 Further work

We see several ways to further develop and expand this ontology:

- As noted, the ontology currently supports a single EA modeler working on creating the EAM. As in practice modelers often work together in creating a single EAM, the ontology must be expanded, and the theory of modeling adapted, to allow for multiple EA modelers.
- The ontology can be used as-is to analyze an actual EA model, preferably involving both the EA modeler and multiple EAM consumers. Such an analysis will provide a check for application validity. We expect that it will also provide insights in what aspects of the EA model contribute to a greater or lesser extent to its quality, and what measures could be taken to guard or improve model quality.
- The ontology can be expanded with research into the relationships between the different terms.
- Using this ontology, specific EA model quality attributes can be identified and documented.
- Several model quality frameworks, such as CMQF [44], have been proposed. A validated model quality framework can provide a systematic structure to categorize and evaluate quality attributes [37]. The ontology can be employed to assist in validating these frameworks.
- Based on the theory of modeling that arose in the drafting of the ontology, it is possible to create modeling guidelines for EA modelers that will positively impact model quality.
- The ontology can be employed in research into RoME, as it provides (for the domain of Enterprise Architecture) an extra footing for both the definition of modeling effort, and for the determination what value an EA model brings.
- The ontology has been drafted for the domain of Enterprise Architecture; however, we believe that it can readily be expanded to envelop any sort of conceptual model.

We conclude by noting that we believe that every bit of additional work will uncover new and rich avenues of exploration, each of which will directly or indirectly lead to tangible improvements in the understanding of modeling and the application of models. This has the potential of significantly improving the modeling practices of tens of thousands of professionals the world over.

Acknowledgements We like to thank Tiago Prince Sales for his feedback regarding the application of OntoUML in underpinning the presented ontology. We furthermore thank Edzo Botjes, Marco Dumont and Neil Kemp for providing additional subject matter expertise as we were constructing the ontology. We would also like to thank the anonymous reviewers. Their feedback has resulted in many improvements to the original article, for which we are grateful.

Funding Open access funding provided by TU Wien (TUW).

References

1. Abraham, R., et al.: Can boundary objects mitigate communication defects in enterprise transformation? findings from expert interviews. In: Jung, R., Reichert, M. (eds) Proceedings of the 5th International Workshop on Enterprise Modelling and Information Systems Architectures, EMISA 2013, St. Gallen, Switzerland, September 5–6, 2013, vol. 222. Lecture Notes in Informatics. Gesellschaft für Informatik Bonn, Germany, pp. 27–40 (2013) ISBN: 978-3-88579-616-9. https://dl.gi.de/handle/20.500.12116/17238
2. Abraham, R., Aier, S., Winter, R.: Crossing the line: overcoming knowledge boundaries in enterprise transformation. In: Business and Information Systems Engineering 57.1, pp. 3–13 (2015). https://doi.org/10.1007/s12599-014-0361-1
3. Apostel, L.: Towards the formal study of models in the non-formal sciences. In Synthese. Int. J. Epistemol. Methodol. Philos. Sci. **12**, 125–161 (1960)
4. Apostel, L.: Towards the formal study of models in the non-formal sciences. In: The Concept and the Role of the Model in Mathematics and Natural and Social Sciences: Proceedings of the Colloquium Sponsored by the Division of Philosophy of Sciences of the International Union of History and Philosophy of Sciences organized at Utrecht, January 1960, by Hans Freudenthal (pp. 1–37). Springer, Dordrecht (1961) ISBN: 978-94-010-3667-2. https://doi.org/10.1007/978-94-010-3667-2_1
5. Architectural Coordination of Enterprise Transformation: In: Proper, H.A., et al. (eds.) Architectural Coordination of Enterprise Transformation. The Enterprise Engineering Series. Springer, Heidelberg (2018) 978-3-319-69583-9. doi: https://doi.org/10.1007/978-3-319-69584-6
6. Artifact. In: Merriam-Webster. https://www.merriam-webster.com/dictionary/artifact (visited on 04/04/2023)
7. Benevides, A.B., Guizzardi, G.: A model-based tool for conceptual modeling and domain ontology engineering in OntoUML. In: International Conference on Enterprise Information Systems, vol. 24. Lecture Notes in Business Information Processing, pp. 528–538. Springer, Berlin, Heidelberg (2009). ISBN: 978-3-642-01347-8. https://doi.org/10.1007/978-3-642-01347-8_44
8. Berman, J., Smyth, R.: Conceptual frameworks in the doctoral research process: a pedagogical model. Innovat. Educ. Teach. Int. **52**(2), 125–136 (2015). https://doi.org/10.1080/14703297.2013.809011. (issn: 1470-3297.)
9. Bjeković, M., Proper, H.A., Sottet, J.-S.: Embracing pragmatics. In: Yu, E. et al. (eds) Conceptual Modeling, pp. 431–444. Springer. ISBN:978-3-319-12206-9 (2014)
10. Dahlberg, I.: A referent-oriented, analytical concept theory for INTERCONCEPT. Int. Classif. **5**(3), 142–151 (1978). https://doi.org/10.5771/0943-7444-1978-3-142. (issn: 0340-0050)
11. De Meyer, P., Claes, J.: An overview of process model quality literature: the comprehensive process model quality framework (Unpublished Work) (2018). https://arxiv.org/abs/1808.07930
12. De Nicola, A., Missikoff, M.: A lightweight methodology for rapid ontology engineering. Commun. ACM **59**(3), 79–86 (2016). https://doi.org/10.1145/2818359
13. Goal. Cambridge Dictionary. https://dictionary.cambridge.org/dictionary/english/goal(visited on 08/29/2022)
14. Goal. Merriam-Webster. https://www.merriam-webster.com/dictionary/goal#synonyms (visited on 07/27/2022)
15. Greefhorst, D., Proper, H.A.: Architecture principles—the cornerstones of enterprise architecture. In: The Enterprise Engineering Series. Springer, Heidelberg (2011). ISBN: 978-3-642-20278-0. https://doi.org/10.1007/978-3-642-20279-7
16. Guarino, N., Guizzardi, G.: We need to discuss the relationship: revisiting relationships as modeling constructs. In: Advanced Information Systems Engineering, pp. 279–294. Springer. ISBN: 978-3-319-19069-3. https://doi.org/10.1007/978-3-319-19069-3_18
17. Guizzardi, G., Proper, H.A.: On understanding the value of domain modeling. In: Guizzardi, G., et al. (eds) Proceedings of 15th International Workshop on Value Modelling and Business Ontologies (VMBO 2021), Bolzano, Italy, 2021, vol. 2835. CEUR Workshop Proceedings. CEURWS. org, 2021. http://ceur-ws.org/Vol-2835/paper6.pdf
18. Guizzardi, G.: Ontological Foundations for Structural Conceptual Models. Thesis (2005). https://research.utwente.nl/en/publications/ontological-foundations-for-structural-conceptual-models
19. Guizzardi, G., et al.: Towards ontological foundations for conceptual modeling: the unified foundational ontology (UFO) story. Appl. Ontol. **10**(3–4), 259–271 (2015). https://doi.org/10.3233/ao-150157
20. Guizzardi, G., et al.: UFO: unified foundational ontology. Appl. Ontol. **17**(1), 167–210 (2022). https://doi.org/10.3233/ao-210256
21. Guo, H., Gao, S.: Achieving alignment by means of EA artifacts. In: Prince Sales, T., et al (eds) Enterprise Design, Operations, and Computing. EDOC 2022 Workshops, vol. 466, pp. 166–179. Springer, Cham. (2023). ISBN: 978-3-031-26885-4. https://doi.org/10.1007/978-3-031-26886-1_10
22. Harel, D., Rumpe, B.: Meaningful modeling: what's the semantics of semantics? Computer **37**(10), 64–72 (2004). https://doi.org/10.1109/MC.2004.172
23. Hestenes, D.: Modeling theory for math and science education. In: Lesh, R., et al. (eds) Modeling Students' Mathematical Modeling Competencies: ICTMA 13, pp. 13–41. Springer, Boston (2010). Chap. 3. ISBN: 978-1-4419-0561-1. https://doi.org/10.1007/978-1-4419-0561-1_3
24. Hjørland, B.: Concept theory. J. Am. Soc. Inf. Sci. Technol. **60**(8), 1519–1536 (2009). https://doi.org/10.1002/asi.21082. (issn:1532-2882.)
25. IEEE. Recommended Practice for Architectural Description of Software Intensive Systems. Technical report IEEE P1471:2000, ISO/IEC 42010:2007. Piscataway, New Jersey: IEEE Explore, Los Alamitos, California (2000)
26. International Organization for Standardization. Software engineering-Software product Quality Requirements and Evaluation (SQuaRE)-Data quality model. Standard (2008). https://www.iso.org/obp/ui/#iso:std:iso-iec:25012:ed-1:v1:en
27. International Organization for Standardization. Terminology work and terminology science—Vocabulary. Standard 1, p. 46. Geneva (2019). https://www.iso.org/obp/ui/#iso:std:iso:1087

28. International Organization for Standardization. Systems and software engineering-Architecture description. Standard. Geneva (2022). https://www.iso.org/obp/ui/#iso:std:isoiec-ieee:42010
29. International Organization for Standardization. Terminology work—Principles and methods. Standard 1, p. 80. Geneva (2022). https://www.iso.org/obp/ui/#iso:std:iso:704:ed-4:v1:en
30. Kendall, E.F., McGuinness, D.L.: Ontology Engineering. Synthesis Lectures on the Semantic Web: Theory and Technology, p. 102 (2019). ISBN: 9781681733098. https://doi.org/10.2200/S00834ED1V01Y201802WBE018
31. Kenyon, G.N., Sen, K.C.: The perception of quality. Springer, London, pp. 265. (2015) https://doi.org/10.1007/978-1-4471-6627-6
32. Klepousniotou, E., et al.: Not all ambiguous words are created equal: an EEG investigation of homonymy and polysemy. In: Brain and Language 123.1, pp. 11–21 (2012). https://doi.org/10.1016/j.bandl.2012.06.007
33. Kotusev, S., Kurnia, S., Dilnutt, R.: Enterprise architecture artifacts as boundary objects: an empirical analysis. Inf. Softw. Technol. **155**, 107108 (2023). https://doi.org/10.1016/j.infsof.2022.107108
34. Krogstie, J.: Quality in Business Process Modeling, p. 250. Springer, Cham (2016)978-3-319-42510-8. doi: https://doi.org/10.1007/978-3-319-42512-2
35. Lankhorst, M.M., et al.: Enterprise Architecture at Work-Modelling. Communication and Analysis. Springer, Heidelberg (2005)3-540-24371-2
36. Lankhorst, M.M., et al.: Enterprise Architecture at Work—Modelling, Communication and Analysis. In: 4th. the Enterprise Engineering Series. Springer, Heidelberg (2017). ISBN: 978-3-662-53932-3. https://doi.org/10.1007/978-3-662-53933-0
37. Lindland, O.I., Sindre, G., Solvberg, A.: Understanding quality in conceptual modeling. IEEE Softw. **11**(2), 42–49 (1994). https://doi.org/10.1109/52.268955
38. Mahr, B.: On the epistemology of models. In: Abel, G., Conant, J. (eds) Rethinking Epistemology, pp. 301–352. De Gruyter, Berlin (2011). ISBN: 9783110253573. https://doi.org/10.1515/9783110253573.301
39. Mayr, H.C., Thalheim, B.: The triptych of conceptual modeling. Softw. Syst. Model. **20**(1), 7–24 (2020). https://doi.org/10.1007/s10270-020-00836-z
40. Menzel, C.: Possible worlds. In: Zalta, E.N., Nodelman, U. (eds) The Stanford Encyclopedia of Philosophy. Winter 2022. Metaphysics Research Lab, Stanford University (2022). https://plato.stanford.edu/archives/win2022/entries/possible-worlds/
41. Mingers, J., Mutch, A., Willcocks, L.: Critical realism in information systems research. MIS Q. **37**(3), 795–802 (2013). https://doi.org/10.25300/MISQ/2013/37:3.3. (**issn: 0276-7783.**)
42. Moore, J.W.: The logic of definition. Technical note. Defence Research and Development Canada, Toronto (2009). https://apps.dtic.mil/sti/citations/ADA504542
43. Muller, P.-A., et al.: Modeling modeling modeling. Softw. Syst. Model. **11**(3), 347–359 (2012). https://doi.org/10.1007/s10270-010-0172-x. (issn: 1619-1374.)
44. Nelson, H.J., et al.: A conceptual modeling quality framework. Softw. Qual. J. **20**(1), 201–228 (2011). https://doi.org/10.1007/s11219-011-9136-9. (issn: 1573-1367.)
45. Nickerson, R.C., Boyd, D.W.: The use and value of models in decision analysis. Oper. Res. **28**(1), 139–155 (1980). https://doi.org/10.1287/opre.28.1.139. (issn: 0030-364X)
46. Objective. Cambridge Dictionary. https://dictionary.cambridge.org/dictionary/english/objective (visited on 08/29/2022)
47. Objective. Merriam-Webster. https://www.merriam-webster.com/dictionary/objective (visited on 08/28/2022)
48. Pareto, L., Eriksson, P., Ehnebom, S.: Architectural descriptions as boundary objects in system and design work. In: Petriu, D.C., Rouquette, N., Haugen, Ø. (eds.) Model Driven Engineering Languages and Systems, pp. 406–419. Springer, Berlin, Heidelberg (2010)
49. Proper, H.A.: On model-based coordination of change in organizations. In: Aier, S., Rohner, P., Schelp, J. (eds) Engineering the Transformation of the Enterprise: A Design Science Research Perspective. Springer, Heidelberg, pp. 79–98 (2021), ISBN: 978-3-030-84655-8. https://doi.org/10.1007/978-3-030-84655-8_6
50. Proper, H.A., Guizzardi, G.: On domain conceptualization. In: Aveiro, D., et al. (eds) Advances in Enterprise Engineering XIV—10th Enterprise Engineering Working Conference, EEWC 2020, Bozen-Bolzano, Italy, September 28, October 19, and November 9–10, 2020, Revised Selected Papers, vol. 411. Lecture Notes in Business Information Processing, pp. 49–69. Springer, Heidelberg (2021). ISBN: 978-3-030-74195-2. https://doi.org/10.1007/978-3-030-74196-9_4
51. Proper, H.A., Guizzardi, G.: Modeling for enterprises-Let's Go to RoME ViA RiME. In: PoEM 2022 Forum Proceedings, vol. 3327. CEURWS. org, pp. 4–15 (2022). https://ceur-ws.org/Vol-3327/paper02.pdf
52. Proper, H.A., Guizzardi, G.: On domain conceptualization. Conference Paper (2020). https://doi.org/10.1007/978-3-030-74196-9_4
53. Purpose. Cambridge Dictionary. https://dictionary.cambridge.org/dictionary/english/purpose (visited on 08/28/2022)
54. Purpose. Merriam-Webster. https://www.merriam-webster.com/dictionary/purpose (visited on 08/28/2022)
55. Recanati, F.: Domains of discourse. Linguist. Philos. **19**(5), 445–475 (1996)
56. Rothenberg, J.: The nature of modeling. In: Widman, L.E., Loparo, K.A., Nielsen, N.R. (eds) Artificial Intelligence, Simulation and Modeling, pp. 75–92. Wiley (1989). (Chap. 3). ISBN: 0471605999
57. Saenz, O., et al.: Defining enterprise systems engineering. Int. J. Ind. Syst. Eng. **4**(5), 483–501 (2009). https://doi.org/10.1504/IJISE.2009.024155. (issn: 1748-5037.)
58. Sandkuhl, K., et al.: From expert discipline to common practice: a vision and research agenda for extending the reach of enterprise modeling. Bus. Inf. Syst. Eng. **60**(1), 69–80 (2018). https://doi.org/10.1007/s12599-017-0516-y. (issn: 1867-0202.)
59. Schoonderbeek, J: Quality attributes of enterprise architecture models. Bachelor's thesis. NOVI university of applied sciences (2020). https://doi.org/10.13140/RG.2.2.10108.67201
60. Schoonderbeek, J.: Toward an ontology for EA modeling and EA model quality. Master's thesis. Antwerp Management School (2023). https://doi.org/10.5281/zenodo.7899004
61. Schulenklopper, J., Rommes, E.: Why they just don't get it: communicating about architecture with business stakeholders. IEEE Softw. **33**(3), 13–19 (2016). https://doi.org/10.1109/MS.2016.67
62. Seidl, M., et al.: UML @ classroom: an introduction to object-oriented modeling. In: Undergraduate Topics in Computer Science. Springer, Cham (2015). ISBN: 978-3-319-12742-2. https://doi.org/10.1007/978-3-319-12742-2
63. Stachowiak, H.: Allgemeine Modelltheorie. Springer, Wien, 494 S. (1973) ISBN: 3-211-81106-0
64. Star, S.L., Griesemer, J.R.: Institutional ecology, 'translations' and boundary objects: amateurs and professionals in Berkeley's museum of vertebrate zoology 1907–39. Soc. Stud. Sci. **19**(4), 387–420 (1989). https://doi.org/10.1177/030631289019003001
65. Steiner, C.M., Albert, D.: Validating domain ontologies: a methodology exemplified for concept maps. Cogent. Educ. **4**(1), 39 (2017). https://doi.org/10.1080/2331186X.2016.1263006. (issn: 2331-186X.)
66. Talha, M.: Total quality management (TQM): an overview. Bottom Line **17**(1), 15–19 (2004). https://doi.org/10.1108/08880450410519656. (issn: 0888-045X.)
67. Thalheim, B. : The theory of conceptual models, the theory of conceptual modelling and foundations of conceptual modelling. In:

Embley, D.W., Thalheim, B. (eds) Handbook of Conceptual Modeling: Theory, Practice, and Research Challenges, pp. 543–577. Springer, Heidelberg (2011). Chap. 17. ISBN: 978-3-642-15865-0. https://doi.org/10.1007/978-3-642-15865-0_17
68. The Open Group. The TOGAF Standard, 10th Edition—Introduction and Core Concepts. Zaltbommel: Van Haren, p. 91 (2022). ISBN: 978-94-018-0860-6. https://publications.opengroup.org/standards/togaf/specifications/c220
69. The Open Group. ArchiMate 3.2 Specification. Van Haren (2023). ISBN: 978-94-018-0956-6
70. The Open Group. The TOGAF Standard, 10th Edition. Accessed on February 1st, 2023. https://www.opengroup.org/togaf/10thedition
71. Timm, F., et al.: Towards a quality framework for enterprise architecture models. In: Quantitative Approaches to Software Quality 2017, vol. 38. Proceedings of the 5th International Workshop on Quantitative Approaches to Software Quality, pp. 14–21 (2017). http://ceur-ws.org/Vol-2017/paper04.pdf
72. The Open Group: TOGAF Version 9-The Open Group Architecture Framework. Van Haren Publishing, Zaltbommel (2009)978-90-8753-230-7
73. Villalón, M.P.: OOPS!-OntOlogy Pitfall Scanner!-Pitfall Catalogue (2021). https://oops.linkeddata.es/catalogue.jsp (visited on 03/04/2023)
74. Zhou, Z., et al.: A systematic literature review on enterprise architecture visualization methodologies. IEEE Access **8**, 96404–96427 (2020). https://doi.org/10.1109/ACCESS.2020.2995850

Jan A. H. Schoonderbeek Jan Schoonderbeek is a practising enterprise architect. At the age of 56 he obtained his MSc degree in Enterprise IT Architecture at Antwerp Management School. Originally trained as an aeronautics engineer at the Delft University of Technology, he went on to become a Linux/Unix engineer, then infrastructure architect, as well as architecture coach/trainer. His main research interest lies in enterprise architecture (EA) modeling, and the way in which EA models serve an organization. Jan currently works on a PhD on EA model quality.

Henderik A. Proper Henderik A. Proper, Erik for friends, is Full Professor in Enterprise and Process Engineering in the Business Informatics Group at the TU Wien. Erik has a mixed background, covering a variety of roles in both academia and industry. His core research drive is the development of theories that work. In other words, Erik focuses on research that leads to results that have both theoretical rigor and practical relevance. His general research interest concerns the foundations and applications of domain modeling; in particular in the context of enterprises. Over the past 20 years, he has applied this research drive and general research interest toward the further development of the field of enterprise engineering, and enterprise modeling in particular. Presently, Erik is vice-chair of the IFIP 8.1 working group, while also being the representative for the Netherlands in IFIP's TC8 technical committee. He is also the Stellvertretender Sprecher of the EMISA working group of the German Computer Science Society (Gesellschaft für Informatik), as well as a member of the management team of the Enterprise Engineering Network.