# Enterprise Modelling in the Age of Digital Transformation

Bas van Gils, Henderik A. Proper

# Enterprise Modelling in the Age of Digital Transformation

Bas van Gils[1] and Hend*erik* A. Proper[2,3][0000−0002−7318−2496]

[1] Strategy Alliance, Lelystad, the Netherlands
[2] Luxembourg Institute of Science and Technology, Luxembourg
[3] University of Luxembourg, Luxembourg
bas.vangils@strategy-alliance.com, e.proper@acm.org

**Abstract.** The digital transformation forces enterprises to change. In addition, the notion of economic exchange, core to the economy, has shifted from following a goods-dominant logic to a service-dominant logic, putting the focus on continuous *value co-creation* between providers and consumers. These trends drive enterprises to transform continuously.
During enterprise transformations, *coordination* among the stakeholders involved is key. Shared understanding, agreement, and commitment, is needed on topics such as: the overall strategy of the enterprise, the current affairs of the enterprise and its context, as well as the ideal future affairs. Models, and ultimately enterprise modelling languages and frameworks, are generally seen as an effective way to *enable* such (informed) coordination. To this end, different languages and frameworks have been developed, including ArchiMate.
ArchiMate, which has evolved to become a widely accepted industry standard, was developed at a time where the digital transformation was not yet that noticeable. At that the time, the focus was more on consolidation and optimisation. As such, it is logical to expect that the existing ArchiMate language may require some "updates" to be ready for digital transformations.
The objective of this paper is therefore threefold: (1) posit, based on practical experiences and insights, key challenges which the digital transformation puts on enterprise (architecture) modelling languages, (2) assess to what extent ArchiMate meets these challenges, and (3) provide suggestions on how to possibly improve ArchiMate to better meet these challenges.

**Keywords:** Enterprise Modelling · Digital Transformation · ArchiMate

## 1 Introduction

Most modern day enterprises find themselves confronted with the challenge of dealing with digital transformations. Where IT originally was a mere supportive tool for administrative purposes, it is safe to say that nowadays IT has become an integral part of an organisation's primary processes. Merely considering the *alignment* of *business* and *IT* [21] no longer suffices. The difference between business and IT is increasingly fading; they have been "fused" into one. Companies such as Amazon, AirBnB, Uber, Netflix, Spotify, Bitcoin, etcetera, illustrate how IT and business have indeed become

fused. The CEO of a major bank can even be quoted as stating "*We want to be a tech company with a banking license*" [20].

In addition, marketing sciences [51, 16, 25, 52] suggest that the notion of economic exchange, core to the economy, has shifted from following a goods-dominant logic to a service-dominant logic. While the former focuses on tangible resources to produce goods and embeds value in the transactions of goods, the latter concentrates on intangible resources and the creation of value in relation with customers. Service-dominance puts the continuous *value co-creation* between providers and consumers at the core. For instance, in the airline industry, jet turbine manufacturers used to follow a classical goods-dominant logic by selling turbines to airlines. However, since airlines are not interested in *owning* turbines, but rather in the realisation of *airtime*, manufacturers nowadays sell airtime to airlines instead of jet turbines. *Value co-creation* is shaping up as a key design concern for modern day enterprises.

We consider the trends of *business-IT fusion* and the *shift to value co-creation*, as being the key challenges to enterprises (be they companies, governmental agencies, or organisations) which aim to thrive (or at least survive) in the digital transformation of society. As a result of these intertwined, and mutually amplifying, trends, enterprises are more than ever confronted with a need to transform.

During any enterprise transformation, *coordination* among the key stakeholders and the projects / activities that drive the transformations is essential [40]. A shared understanding, agreement, and commitment, is needed on (1) what the overall strategy of the enterprise is, (2) the current affairs of the enterprise, i.e. the current situation, as well as the relevant history leading up to it, and possible trends towards the future, (3) the current affairs of the context of the enterprise, and (4) what (given the latter) the ideal future affairs of the enterprise are. Borrowing the terminology from architecture frameworks such as TOGAF [48], this refers to the development of a shared vision, a baseline architecture, and a target architecture, respectively.

Models, and ultimately enterprise (architecture) modelling languages and framework, are generally considered as an effective way to support such (informed) coordination. Many languages and frameworks have indeed been suggested as a way to create and capture a shared understanding of the desired future affairs. Examples, include BPMN [14], UML [28], ArchiMate [4, 24], 4EM [46], and MERODE [47]. It appears[4] that ArchiMate [4, 24] is rapidly becoming a / the leading industry standard for enterprise architecture modelling and has, as such, a key role to play in the coordination of [40] enterprise transformations.

When ArchiMate was developed, the digital transformation challenges were not yet that noticeable. At that time, the focus was more on consolidation and optimisation. As such, it is logical to expect that the existing ArchiMate language may require some "updates" to be truly ready for the digital transformation, and the emerging focus on value co-creation. The objective of this paper is therefore threefold: (1) define some of the challenges that the digital transformation puts on enterprise architecture modelling

[4]The support for this claim lies in the steady growth of the number of certified professionals `http://archimate-cert.opengroup.org/certified-individuals` as well as the popularity of the ArchiMate topic on Google trends `https://trends.google.com/trends/explore?date=all&q=archimate`.

languages, (2) assess to what extent ArchiMate meets these challenges, and (3) provide suggestions on how to possibly improve ArchiMate to better meet the challenges of digital transformations. These topics will be covered in Sections 2, 3 and 4 respectively.

## 2 Challenges for Enterprise Modelling

In this Section, we identify some of the key challenges on enterprise (architecture) modelling in the context of digital transformations. These challenges are also based on our own experience in using ArchiMate in practice,[5] as well as teaching the language to practitioners[5] and University students.[6]

In this paper, we consider the *digital transformation of an enterprise* to be any *enterprise transformation* which has a major impact on the digital resources and capabilities of the enterprise, where we define *enterprise transformation* to be a coordinated effort that changes the architecture of an enterprise.

The identified challenges have been grouped in three classes. First, we discuss challenges pertaining to the general expressiveness of the modelling language used. Since the digital transformation and value co-creation trends push for further specialisation and domain specificity of modelling languages, the second class of challenges concerns the need to be able to manage the resulting spectrum of modelling concepts. Finally, the third class concerns the earlier made observation that the digital transformation fuels the speed of change in organisations and their enterprises.

**Expressiveness of the modelling language**

In traditional views on enterprise architecture, it was more or less assumed that objects were either passive (operand) or active (operant), but not both, for their entire life. This simplification might indeed have worked in former times. However, in the context of the digital transformation, this simplification becomes increasingly difficult to uphold, especially since digital objects may (aid to) create other digital objects.

Key is, that it is natural for the same objects to play different roles in the course of time, or even in parallel. An enterprise architecture modelling language used in digital transformations should therefore support this plurality of the roles played by objects:

**Challenge 1:** *Objects should be allowed to play operand and operant roles.*

Digital transformations also result in an increased reliance on the quality of information in terms of being *aware* of the level of (in)correctness at which it represents the world around us. This makes it increasingly important to remain aware of, and explicitly capture, the distinction between elements *in* the real world and the information that (is assumed to) *refer to* those real world elements. For example, in terms of a clear distinction between *business objects* as they exist in the real world, and *business information objects* that represent information *about* the former. Enterprise (architecture) modelling languages should, therefore, also clearly reflect such a distinction:

[5] Involving Dutch public institutions, international fund management institutions, banks and lease companies, as well as retail organisations.

[6] Typically involving groups of full-time and / or part-time (i.e. practitioners) MSc students from e.g. Antwerp Management School, TIAS business school, Radboud University, Technical University of Vienna, and the University of Luxembourg.

**Challenge 2:** *Clear separation between objects that represent "things" in the real world, and objects representing information about the real world.*

A consequence of the digital transformation [8] is that we should prepare for new forms of diversity in the work-force, where humans should learn to collaborate closely with digital actors (e.g. agents, robots, etcetera). Modern day enterprise modelling languages should, therefore, have the:

**Challenge 3:** *Ability to deal naturally with the duality of human and digital actors.*

A key aspect in traditional (conceptual) data modelling is the notion of *unique identification*; i.e. the ability to specify how objects in the real world can be distinguished from one another. The need to be able to model this depends on the situation at hand. Not all applications will need it, while in some cases it might even be illegal, e.g. due to privacy considerations. Even when a unique identification mechanism is available there may be limits regarding its precision. In a business network involving multiple partners, one may have to use multiple, partially overlapping, identification mechanisms. Even more, one may not have control over the creation of objects, which may (accidentally or maliciously) end up having the same properties as used in the identification. For enterprise modelling languages, this leads to the following challenge:

**Challenge 4:** *Ability to specify if objects can, should, and / or are allowed, to be uniquely identified, and what the expected reliability is.*

Most enterprise modelling languages do not allow for detailed modalities (mandatory, optional, one-to-one, one-to-many, exclusion, etcetera) on relationships. In general, this has been a deliberate choice by the language designers. In practice, however, this decision becomes increasingly challenged. It has been debated extensively among practitioners – for example in the LinkedIn group for ArchiMate as well as during training and coaching sessions – how useful it would be to be able to specify modalities, in particular in e.g. the context of privacy and security. A typical example would be the four-eyes principle, where two roles must be fulfilled when performing a certain task.

We suggest that, although one should not categorically require architecture models to use modalities on relationships, this should be addable when needed:

**Challenge 5:** *Ability to specify modalities on relationships.*

Several studies [51, 26, 16] observe a fundamental shift from, what they call, a goods-dominant logic to a service-dominant logic. While the former focuses on the production of goods, the latter concentrates on the delivery of services using resources and / or goods in doing so. These studies motivate this shift by observing that it is ultimately the customer who attributes value to a good or a service. Goods and services, "at rest", only have a potential value to a customer. The actual value is experienced when the resources / goods are actually *used* by the customer to some purpose.

The *digital transformation* not only brings about a new wave of digital services, it also acts as an enabler that allows providers of goods and service to better optimise the co-creation of value with their customers. Leading to the challenge for enterprise (architecture) modelling languages:

**Challenge 6:** *Ability to capture (potential) value(s) of products and services, and how this results in value co-creation between providers and consumers of services by way of resource integration.*

Given the speed of technological developments that drive digital transformations, it is increasingly important for organisations to be aware of the design choices that shape the essence of their activities, as well as choices with regards to their implementation in terms of e.g. different platforms, (business proces) outsourcing, and technologies.

For enterprise modelling languages, this means that one should be able to express the design of the enterprise (including its use of information technology) at different levels of specificity with regards to implementation decisions, as well as enable the capturing of the associated design decisions and their motivation [34, 33].

**Challenge 7:** *Capture design decisions and their motivation, at different levels of specificity with regards to implementation decisions.*

**Managing the spectrum of modelling concepts**

An enterprise modelling language typically features a rich set of modelling concepts. As a natural consequence of the use of such a language, and as a corollary to the law of increasing entropy, there is a tendency to continue adding concepts to modelling languages without cleaning up concepts and relations that are infrequently used [5].

The digital transformation, due to its deep impact and multi-facetness, is likely to further fuel the entropic forces. Some of the challenges listed above, already point towards a desire to extend existing modelling languages. At the same time, an ever increasing set of modelling concepts will lead to a modelling language that will be hard to learn and master [27, 22]. This leads to the following challenge:

**Challenge 8:** *A way to manage the set of modelling concepts, balancing the needs of domain, and purpose, specificity, the need for standardisation, and comprehensibility of the modelling language.*

Enterprise modelling languages typically involve different abstraction layers. Examples include the business, application and technology layer as used in ArchiMate [24], the business, information systems and technology layer from TOGAF [48], the business, information, information systems, and technology infrastructure columns from IAF [53], as well as the conceptual, logical, and physical layers of the same.

We observe in practice (both in using such frameworks, as well as teaching about them) that confusion about the precise scoping of the used abstractions exists. In this regard, one can even distinguish changes in the interpretation of the business, application, and technology layer, from the earlier version(s) of ArchiMate, where the technology layer was purely intended as the (IT) technological *infrastructure*, to the current interpretation, where it has evolved to include the entire (IT) technological implementation.

In general, one could say that abstraction layers result from the *design philosophy* underlying the specific framework. In this paper, we do not aim to take a specific position on which *design philosophy* would be best. However, we do argue that it is important that modelling frameworks must provide clear and consistent abstractions:

**Challenge 9:** *Provide a structure that allows for a consistent use of abstractions across relevant aspects of the enterprise.*

Enterprise (architecture) models play an increasingly important role. When changing an enterprise, models are used to capture the current affairs, as well as articulate different possible future affairs. Even more, nowadays it is quite common that models are even part of the "running system", in the sense that they are an artefact that drives / guides day-to-day activities. This includes e.g. work-flow models and business rules.

This makes it important that enterprise models also capture their meaning in a way that is understandible to the model's audience. We therefore posit that a conceptual model should be grounded in the terminology as it is actually *used* (naturally) by the people involved in / with the modelled domain. We also see this as a key enabler for the transferability of models across time and among people, in particular in situations where the model needs to act as a *boundary object* [3].

Most existing enterprise modelling languages (e.g. process models, goal models, value models, architectural models, etcetera), only offer a "boxes and lines" based representation, which, by its very nature only provide a limited linkage to the (natural) language as used by the model's audience. In general, the only link in this regard are the names used to label the "boxes". Relationships are replaced by generic graphical representations in terms of arrows and lines capturing relations such as "assigned to", "part of", "realises", "aggregates", "triggers". While these notational styles enable a more compact representation of models, they offer no means to provide a "drill down", or "mouse over", to an underlying grounding in terms of well verbalised fact types that capture, and honour, the original natural (language) nuances. They leave no room for situation specific nuance, or more explicit capturing of the meaning of the models a way that is understandible to the model's audience (beyond engineers). The challenge therefore is:

**Challenge 10:** *How to ground enterprise models in terms of natural language like verbalisations, without loosing the advantages of having compact notations (as well).*

**Enterprises are in motion**

Modern day enterprises, and their context, are in a constant state of change; they are continuously in motion. As argued before, the digital transformation of society further increases the speed at which enterprises change. As such, it is doubtful, if not unrealistic, to use traditional notions such as "baseline" architecture and "target" architecture. For instance, an enterprise's baseline can not simply be thought of "structural state", but should rather be thought of as a "structural vector" [38, 35]. The rate of change in modern day enterprises is so high, that maintaining models that sufficiently capture the architecture (i.e. the fundamental organisation and the principles guiding design and evolution) from the perspective of involved stakeholders does not seem to be feasible. Hence, it is better to speak about capturing "current affairs", which includes past, and present, change trends, of the enterprise and its environment. This allows one to reason both about "what is" as well as "what was before" and therefore take the history into account when making decisions. This results in the challenge:

**Challenge 11:** *How to capture the motion of an enterprise, it terms of its current and desired affairs.*

## 3 ArchiMate's Readiness for the New Modelling Challenges

In this Section, we discuss to what extent the current version of ArchiMate meets the challenges of digital transformations, as identified in the previous Section.

ArchiMate[7] is a dedicated language for the representation of (enterprise) architecture models that was originally developed by a research consortium involving industrial

[7] At the time of writing, ArchiMate 3.0.1 is available online

and academic partners from the Netherlands. Later, it was adopted by The Open Group as a standard [24, 4]. Its adoption has grown rapidly, both in terms of the *users* of the language, and the *vendors* that deliver software solutions based on this language.

The current version covers six layers (strategy, business, application, technology, physical, as well as implementation & migration) and four aspects (active structure, passive structure, behaviour, and motivation). The core consists of the business / application / technology layers. Services are a key modelling construct in ArchiMate, and are used as a decoupling mechanism between the layers. They are used to specify what an active structure element exposes to its environment and hides the complexity of how the service is actually realised. Services can be used within a layer or across layers.

A second abstraction mechanism in the language is the *specialisation relation*, which is to be interpreted as "is a kind off". Some languages, such as ORM [19] and UML [28], distinguish between a) specialisation, b) generalisation, and c) type / instance relations. In ArchiMate, these are all captured by the same specialisation relation. Using this relation, it is possible to relate generic architecture constructs (e.g. a process pattern) to more specific manifestations (e.g. distinguishing between the regular manifestation of the process, or the one that is followed during times of crisis).

An abstraction mechanism that was introduced in version 3 of the ArchiMate language is the use of *grouping*. Previously, the grouping was a visual construct only, which was intended to show on a view which concepts "belong together" for some reason. Since ArchiMate version 3, the intended meaning is more rigourous: the grouping is said to aggregate the concepts that are in it, and thus functions as a semantic whole. Groupings may be related to other concepts (including other groupings). This makes it particularly well suited to use the grouping as a form of *building blocks* along the lines of the TOGAF standard (e.g. [48, Chapter 37]).

The last mechanism that is relevant to our discussion here is the notion of *cross layer dependencies* [4, Chapter 12]. This has changed significantly between version 2 and 3 of the language. The general idea is that behavioural elements from one layer may be *realised* by (more concrete) behavioural elements in other layers, which appears to express that the more abstract concept is *instantiated* by the more concrete one. Along the same lines, it allows us to specify that a group of elements (i.e. a building block) is realised by another group of elements (another building block).

In the remainder of this Section, we briefly assess ArchiMate in the light of the challenges brought forward by the digital transformation.

**Challenge 1:** *Objects should be allowed to play operand and operant roles.*
The current ArchiMate language does not deal with this at all, due to the strict distinction between *active* and *passive* structure elements. This challenge lies at the heart of the ArchiMate language and has its origin in the fundamental choice to use a linguistic (subject-object-verb) base for modelling.

**Challenge 2:** *Clear separation between objects that represent "things" in the real world, and objects representing information about the real world.*
Currently, there is no clear distinction between the two types of objects, other than the observation that data objects / artefact presumably are about the bits and bytes that represent information. The ArchiMate specification does suggest that the *business object* concept can be specialised but in the default language this has not been done. What the

specification does not mention is that additional relations may also be required in order to present that a *business information object A* is about real world *business object B*.

**Challenge 3:** *Ability to deal naturally with the duality of human and digital actors.*
Here, the ArchiMate language, through its layering, does provide a fair attempt at tackling this challenge since there are different concepts for e.g. actor, information system, and node. Some interesting challenges remain, however. First of all, only (business) actors can be assigned a role in behaviour, other structure elements cannot. A second mismatch lies in the fact that *collaborations* in ArchiMate can only be composed of structure elements from the same layer. This prevents us from specifying, for example, that a human actor and computer actor collaborate to achieve a certain task.

**Challenge 4:** *Ability to specify if objects can, should, and / or are allowed, to be uniquely identified, and what the expected reliability is.*
In ArchiMate, (most) concepts are essentially "types", representing "instances" in the real world. The ArchiMate concepts, representing "things in the real world" have a name to tell one apart from the other. There is no mechanism to specify how the "instances" should be told apart.

**Challenge 5:** *Ability to specify modalities on relationships.*
For this challenge we can be short. ArchiMate has no support for this. Objects are either related, or they are not.

**Challenge 6:** *Ability to capture (potential) value(s) of products and services, and how this results in value co-creation between providers and consumers of services by way of resource integration.*
The ArchiMate language does feature the *value* concept, and it seems possible to model value *co*creation by using the collaboration / interaction concepts. However, value, or even a value stream, is not yet value *co-creation*. As discussed in e.g. [42, 43], representing value co-creation [16] scenarios requires more dedicated modelling constructs.

**Challenge 7:** *Capture design decisions and their motivation, at different levels of specificity with regards to implementation decisions.*
There is limited support in ArchiMate to address this. The actual level of detail at which design decisions remains rather crude. For example, it excludes explicit trade-offs between design alternatives, nor does it capture the actual decision making process and (compensatory and / or non-compensatory [44]) criteria used to make decisions.

**Challenge 8:** *A way to manage the set of modelling concepts, balancing the needs of domain, and purpose, specificity, the need for standardisation, and comprehensibility of the modelling language.*
This challenge is addressed partially by the *extension mechanisms* to tailor the language to local needs, while keeping the core of the language compact. This can be done by specialising existing concepts, or adding properties to existing concepts. Additions to the original version of the language have also been positioned as so-called "extensions", such as the *motivation* extension, and the *implementation & migration* extension.

While it is good that the language indeed supports this, being able to re-use extensions across toolsets of different vendors is not straightforward. Even more, the actual extension mechanism is not really positioned as a key feature in the standard either.

**Challenge 9:** *Provide a structure that allows for a consistent use of abstractions across relevant aspects of the enterprise.*

The latest version of ArchiMate does indeed provide some rudimentary support to tackle these challenges through the *grouping* mechanism. It is now possible to express the fact that one group of concepts (together) realises another group of concepts. This allows the modeller to work from a big picture level to a more detailed level, as well as from a functional level to a more construction-oriented level.

**Challenge 10:** *How to ground enterprise models in terms of natural language like verbalisations, without loosing the advantages of having compact notations (as well).*

ArchiMate has no support for this; neither in the language nor the modelling process (which is non existent). Even more, there is no pre-defined modelling procedure such as ORM's CSDP [19], leaving (in particular novice modellers) to guess how to master ArchiMate's elaborate set of modelling concepts [37].

**Challenge 11:** *How to capture the motion of an enterprise, it terms of its current and desired affairs.*

Support for this challenge is limited. TOGAF and ArchiMate have indeed extended the concepts of "baseline" and "target" architecture into a multi-stage version in terms of "plateaus" towards the future. They allow the modeller to specify multiple points in time as well as capture "alternate realities" (i.e. different potential futures) through the use of *plateaus*. Even more, ArchiMate allows the modeller to link concepts to motivational elements of key stakeholders.

Building a modelling language that supports modellers to consistently solve challenges, and solve them well while keeping in the return on modelling effort ("RoME") in mind, is a difficult task indeed. After listing the modelling challenges for digital transformations, and evaluating the current version of ArchiMate against these challenges, we conclude that, even though ArchiMate has been around for a while, and has a strong conceptual framework, its support for digital transformations can be improved.

## 4 Recommendations for Next Generation EA Modelling

Below, we provide (motivated) recommendations that could overcome (some of) the challenges as discussed above.

**Modular language design** – The set of modelling constructs within the ArchiMate language has grown considerably since its first version, thus fuelling Challenge 8. In meeting this challenge, we suggest that modelling language standards in general should focus primarily on providing a generic core of well-defined modelling concepts. On top of this core, one could then define refinement mechanisms, that can be used to extend / tailor the core to the needs at hand. This may involve both specialisations of the core concepts, as well as e.g. the introduction of (purpose specific / user defined) layers.

As discussed in the previous Section, the use of ArchiMate's *extension mechanism* indeed provides a potential starting point to better manage the resulting set of concepts. The positioning of recent additions to the language as *extensions*, such as the *motivation* extension, and the *implementation & migration* extension, indeed underlines this. When looking at the original architecture of the ArchiMate language [23], there are ample opportunities for further modularisation. Following [23], the core of the language is formed by five key generic "active systems" modelling concepts: *objects*, *service*, *internal behaviour*, *interface* and *internal structure*. All other concepts (including the layers)

are explicitly derived from these in terms of specialisations [23]. In our view this specialisation hierarchy has been left too implicit for far too long. An explicit re-factoring of the current ArchiMate language based on this hierarchy seems long overdue.

In addition, a library of (meta-model) *modules* can be defined, which could potentially even be (re)used across different language cores. For example, a generic *motivation* module could be shared between ArchiMate, 4EM [46] and UML [28].

**Grounding enterprise modelling** – As suggested by Challenge 10, enterprise models should (unless they only serve a temporary "throw away" purpose) include a precise (enough) definition of the meaning of the concepts used in the model.

We posit that, to ensure that a model is understandable to its audience, it should be grounded on an (underlying) model involving verbalisations using the terminology as it is actually *used* (naturally) by the people involved in / with the modelled domain. As exemplified in [6, 7, 50, 37], fact-based models can be used to ground enterprise models expressed in languages such as ArchiMate, system dynamics [45] and BPMN [29], and architecture principles [15]. Fact-based models [19] are created by verbalising the domain to be modelled in terms of elementary facts as structured sentences in natural language. These elementary facts expres (unsplittable) properties of, and relationships between, the objects in the modelled domain. Based on these elementary facts, the conceptual model is then created in terms of fact-types and object-types. An added advantage of using fact-based models is that it also leads to an evidence (in terms of example facts) based approach to modelling.

Grounding ArchiMate on fact-based models would also result in a natural solution to Challenge 1, i.e. the need for objects to be able to play operand and operant roles. When observing objects and expressing their engagements in activities in terms of fact types, one can easily observe objects mixing *passive structure* / *active structure* / *behaviour* roles. Even though we strongly suggest to remain close to the terminology as it is actually *used* by the people involved in / with the modelled domain, we do see the potential benefits of providing guidance in structuring / refining this terminology based on e.g. foundational ontologies [18].

**Adding more semantic precision** – Both Challenge 4 and 5 require the ability to specify more semantic details regarding objects and relations. It would be logical to "import" such mechanisms from these existing languages into ArchiMate. When, grounding enterprise models (e.g., using a fact-based approach as discussed above), then this will also come as a direct consequence. Even though it is not required for architects to specify such constraints in all situations, it is key to provide the ability to do so when required.

**Abstraction layers** – Challenge 2 and 9, are essentially all concerned with the need to "separate concerns", albeit in different ways. As argued before, it is important to ensure a clear and consistent structure of abstraction layers. When looking across different frameworks (e.g. ArchiMate [24], Enterprise Ontology [11], TOGAF [48], IAF [53], and Zachman [54]), we posit that these frameworks use four key mechanisms in creating abstractions (in different dimensions, possibly combining these mechanisms).

*1: Function-construction* – This abstraction mechanism involves making a distinction between, *function* referring to the way a system is intended to function in light of what users, clients, and other stakeholders might deem useful, and *construction* pertaining to the way a system actually functions / is constructed to realise the provided functions.

*2: Informational functioning* – This pertains to different levels of *functioning* [10, 30, 39] of an enterprise in terms of informational support, e.g. leading to a *business* level (the activities conducted that have a direct impact in the socio-economical world), an *informational* level (the information needed / created in the business activities) and a *documental* level (how this information may be laid down in documental concepts). A clear distinction between *business* and *informational* activities, also results in a natural way to deal with Challenge 2; i.e. separating the real world and the informational world.

*3: Infrastructural usage* – This concerns the fact that one system (of systems) can *use* the functions of another system (of systems), where the actual construction of the latter is of no interest to the (designers) of the former, except to the extent of defining service-level agreements.

*4: Implementation abstraction* – This concerns the gradual / stepwise introduction of details of the socio-technical implementation. For example, in IAF [53] this corresponds to the distinction between a conceptual, logical, and physical level, while in TOGAF [48], this corresponds to the level of architectural and logical building blocks.

Making a clear implementation abstraction, also provides a natural way to deal with Challenge 3 pertaining to the duality of human and digital actors. At the highest level of implementation abstraction, one would need to describe the workings of the enterprise independent of the question if it will be implemented with human actors or computerised actors. The immediate next level of implementation abstraction, might then make choices with regards to human / computerised actors explicit, even allowing for mixed scenarios.

Each of the above discussed abstraction mechanisms has a potential added value, also in the context of digital transformation. It is important to note that these abstraction mechanisms should not be thought of as a set of orthogonal dimensions. On the contrary. The *function-construction* mechanism and *information functioning*, or *function-construction* and *infrastructural usage* can easily be mixed. We also do not want to suggest to "prescribe" a specific set of dimensions. We do, however, argue that an enterprise modelling language (framework) should ensure a consistent use of the above mechanisms within one dimension. As discussed in Section 3, ArchiMate seems to have been mixing some of these dimensions in an inconsistent way.

**Accommodate value co-creation** – The increasing focus on value co-creation, resulting from the shift from a goods-dominant logic to a service-dominant logic [51, 16, 25, 52], results in Challenge 6, i.e. how to capture (potential) value(s) of products and services, and how this results in value co-creation between providers and consumers of services by way of resource integration.

ArchiMate already provides *value* concept, and it seems possible to model value *co*creation by using the collaboration / interaction concepts. However, as mentioned before, value, or even a value stream, is not the same as value *co-creation*. How to best express this, is still largely an open question. Some initial work / suggestions, has been presented in [41, 12, 13, 42].

The very nature of value co-creation also requires a shift from (only) architecting the "internals" an an enterprise, to co-architecting the collaboration (including e.g. needed inter-organisational IT platforms) between multiple network partners [9].

In further elaborating the set of needed concepts for value co-creation, our recommendation [36] is to (1) use the provider / customer roles as identified in [17], specialised to more specific co-creation activities taking place within the *provider sphere*, the *joint sphere*, or the *customer sphere*, as a reference model, while (2) using the foundational premises as articulated in [52] as design / architecture principles [15] that will guide the design of service systems for value co-creation, and (3) apply this in the context of real world cases, to gain insight into the actually needed modelling concepts.

**Capturing design motivations** – The current version of ArchiMate does provide a motivation extension. However, as discussed in the previous Section, it does not meet Challenge 7 in a satisfactory way. Separate from the fact that, as suggested above, it would be good if such an extension could be shared between e.g. ArchiMate, 4EM [46] and BPMN [29], the level at which design decisions are captured remains rather crude.

The work as reported in e.g. [32, 31] provides suggestions on how to remedy this. This includes the ability to e.g. capture trade-offs between design alternatives, as well the actual decision making process and the criteria used to make decisions (including e.g. the identification of compensatory and / or non-compensatory [44] criteria).

**Managing constant change** – As discussed in Section 2, the digital transformation requires enterprises to change constantly. This makes it less realistic to capture an enterprise's *current affairs* and / or *desired affairs* in terms of traditional notions such as "baseline" architecture and "target" architecture, or even *plateaux* / *transition* architectures. Even though we observe some ingredients towards solutions for this challenge, we would argue that more research is certainly needed.

In an ideal world, the description of the *current affairs*, would be maintained continuously, preferably in an automated way [35]. Approaches such as e.g. process mining [1], and enterprise cartography [49], indeed provide good starting points.

Architectures capturing the *desired affairs*, also tend to be specified using a rather "instructive" of typical "boxes and lines" diagrams. This does not really invite architects to reflect on what the more *endurable* elements and assumptions, and what the less stable elements and assumptions are. This has also triggered the development of the concept of multi-speed enterprise (IT) architectures [2]. It also resulted in a stronger positioning of e.g. (normative) architecture principles [15] as a way to complement the "instructive" style (the "boxes and lines" diagrams) by a more "directional" / "regulative" perspective.

## 5 Conclusion & Further Research

In this paper, we presented key challenges which the digital transformation puts on enterprise (architecture) modelling languages. These challenges are based on practical experiences and insights from the field of enterprise architecture.

We then assessed the extent to which the current version of ArchiMate meets these challenges. The conclusion was that ArchiMate does not yet fully cover all of the identified challenges. This can be explained by the fact that ArchiMate was developed at a time when the digital transformation was not yet that dominant.

We then provided suggestions on how to possibly improve ArchiMate to better meet the challenges of digital transformations. In further research, we intend to further elab-

orate these suggestions, in particular with the aim of finding strategies that work in real world practice.

## References


1. van der Aalst, W.M.P.: Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer (2011)
2. Abraham, R., Aier, S., Winter, R.: Two Speeds of EAM – A Dynamic Capabilities Perspective. In: Aier, S., Ekstedt, M., Matthes, F., Proper, H.A., Sanz, J. (eds.) Proc. of the 7th Workshop on Trends in Enterprise Architecture Research (TEAR 2012) and the 5th Working Conf. on Practice-driven Research on Enterprise Transformation (PRET 2012). Held at The Open Group Conf. LNBIP, vol. 131, pp. 111–128. Springer (October 2012)
3. Abraham, R., Niemietz, H., de Kinderen, S., Aier, S.: Can boundary objects mitigate communication defects in enterprise transformation? Findings from expert interviews. In: Jung, R., Reichert, M. (eds.) Proc. of the 5th Int. Workshop on Enterprise Modelling and Information Systems Architectures (EMISA 2013), St. Gallen, Switzerland. LNI, vol. 222, pp. 27–40. Gesellschaft für Informatik (2013)
4. Band, I., Ellefsen, T., Estrem, B., Iacob, M.E., Jonkers, H., Lankhorst, M.M., Nilsen, D., Proper, H.A., Quartel, D.A.C., Thorn, S.: ArchiMate 3.0 Specification. The Open Group (2016)
5. Bjeković, M., Proper, H.A., Sottet, J.S.: Enterprise Modelling Languages – Just Enough Standardisation? In: Shishkov, B. (ed.) Business Modeling and Software Design, Third Int. Symposium, BMSD 2013, Noordwijkerhout, the Netherlands, July 8-10, 2013, Revised Selected Papers. LNBIP, vol. 173, pp. 1–23. Springer (2014)
6. van Bommel, P., Buitenhuis, P.G., Hoppenbrouwers, S.J.B.A., Proper, H.A.: Architecture Principles – A Regulative Perspective on Enterprise Architecture. In: Reichert, M., Strecker, S., Turowski, K. (eds.) Proc. of the 2nd Int. Workshop on Enterprise Modelling and Information Systems Architectures (EMISA 2007), St. Goar am Rhein, Germany. pp. 47–60. No. 119 in LNI, Gesellschaft für Informatik (2007)
7. van Bommel, P., Hoppenbrouwers, S.J.B.A., Proper, H.A., van der Weide, T.P.: On the Use of Object-Role Modeling For Modeling Active Domains, pp. 123–145. Research Issues in System Analysis and Design, Databases and Software Dev., IGI Publishing (2007)
8. Brown, S.: The new diversity: working with Nonhumans. IEEE Computer **50**(3), 90–95 (April 2017)
9. Chew, E.K.: iSIM: An integrated design method for commercializing service innovation. Information Systems Frontiers **18**(3) (2016)
10. Dietz, J.L.G.: Enterprise Ontology – Theory and Methodology. Springer (2006)
11. Dietz, J.L.G., Hoogervorst, J.A.P.: Enterprise Ontology and Enterprise Architecture – how to let them evolve into effective complementary notions. GEAO Journal of Enterprise Architecture **1** (2007)
12. Feltus, C., Proper, H.A.: Conceptualization of an Abstract Language to Support Value Co-Creation. In: 12th Conf. on Information Systems Management (ISM'17), Federated Conf. on Computer Science and Information Systems. Prague, Czech Republic (2017)
13. Feltus, C., Proper, H.A.: Towards a Security and Privacy Co-Creation Method. In: IEEE Int. Conf. for Internet Technology and Secured Transactions (ICITST-2017) (2017)
14. Freund, J., Rücker, B.: Real Life BPMN. Camunda (2012)
15. Greefhorst, D., Proper, H.A.: Architecture Principles – The Cornerstones of Enterprise Architecture. Enterprise Engineering Series, Springer (2011)

16. Grönroos, C., Ravald, A.: Service as Business Logic: Implications for Value Creation and Marketing. Journal of Service Man. **22**(1), 5–22 (2011)
17. Grönroos, C., Voima, P.J.: Critical service logic: making sense of value creation and co-creation. Journal of the Academy of Marketing Science **41**(2), 133–150 (2013)
18. Guizzardi, G.: On Ontology, ontologies, Conceptualizations, Modeling Languages, and (Meta)Models. In: Vasilecas, O., Eder, J., Caplinskas, A. (eds.) Databases and Information Systems IV - Selected Papers from the Seventh Int. Baltic Conference, DB&IS 2006, July 3-6, 2006, Vilnius, Lithuania. Frontiers in Artificial Intelligence and Applications, vol. 155, pp. 18–39. IOS Press (2006)
19. Halpin, T.A., Morgan, T.: Information Modeling and Relational Databases. Data Management Systems, Morgan Kaufman, 2nd edn. (2008)
20. Hamers, R.: We want to be a tech company with a banking license (August 2017), `https://tinyurl.com/ycn8l2kh`, last accessed on 25th of January, 2018
21. Henderson, J.C., Venkatraman, N.: Strategic alignment: Leveraging information technology for transforming organizations. IBM Systems J., **32**(1), 4–16 (1993)
22. Krogstie, J., Lindland, O.I., Sindre, G.: Defining Quality Aspects for Conceptual Models. In: Falkenberg, E.D., Hesse, W., Olivé, A. (eds.) Information System Concepts: Towards a consolidation of views – Proc. of the third IFIP WG8.1 conference (ISCO–3). pp. 216–231. Chapman & Hall/IFIP WG8.1, London, UK, Marburg, Germany (March 1995)
23. Lankhorst, M.M., Proper, H.A., Jonkers, H.: The Anatomy of the ArchiMate Language. Int. Journal of Information System Modeling and Design **1**(1), 1–32 (2010)
24. Lankhorst, et al, M.M.: Enterprise Architecture at Work: Modelling, Communication and Analysis. Enterprise Engineering Series, Springer, 4th edn. (2017)
25. Lusch, R.F., Nambisan, S.: Service Innovation: A Service-Dominant Logic Perspective. MIS Quarterly **39**(1), 155–175 (2015)
26. Maglio, P.P., Vargo, S.L., Caswel, N., Spohrer, J.: The Service System is the Basic Abstraction of Service Science. Information Systems and e-business Management **7**(4), 395–406 (2009)
27. Moody, D.L.: The "Physics" of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering. IEEE Transactions on Software Engineering **35**(6), 756–779 (2009)
28. Object Management Group: Unified Modeling Language – Superstructure. Tech. Rep. version 2.4.1, OMG (2010)
29. OMG: Business Process Modeling Notation, V2.0. Tech. Rep. OMG Document Number: formal/2011-01-03, Object Management Group (January 2011)
30. Op 't Land, M., Proper, H.A.: Impact of Principles on Enterprise Engineering. In: Österle, H., Schelp, J., Winter, R. (eds.) Proc. of the 15th European Conf. on Information Systems. pp. 1965–1976. University of St. Gallen, St. Gallen, Switzerland (June 2007)
31. Plataniotis, G., de Kinderen, S., Ma, Q., Proper, H.A.: A Conceptual Model for Compliance Checking Support of Enterprise Architecture Decisions. In: Proc. of the 17th IEEE Conf. on Business Informatics (CBI 2015), Lisbon, Portugal. vol. 1, pp. 191–198 (July 2015)
32. Plataniotis, G., de Kinderen, S., Proper, H.A.: Capturing Design Rationales in Enterprise Architecture: A case study. In: Proc. of the 7th IFIP WG 8.1 working conference on the Practice of Enterprise Modeling (PoEM 2014) (2014)
33. Plataniotis, G., de Kinderen, S., Proper, H.A.: EA Anamnesis: An Approach for Decision Making Analysis in Enterprise Architecture. Int. Journal of Information System Modeling and Design **5**(3), 75–95 (2014)
34. Plataniotis, G., Ma, Q., Proper, H.A., de Kinderen, S.: Traceability and modeling of requirements in enterprise architecture from a design rationale perspective. In: Proc. of the 9th IEEE Int. Conf. on Research Challenges in Information Science (RCIS 2015), Athens, Greece. pp. 518–519 (May 2015)

35. Proper, H.A.: Enterprise Architecture – Informed steering of enterprises in motion. In: Proc. of the 15th Int. Conf. (ICEIS 2013), Angers, France – Revised Selected Papers. LNBIP, vol. 190, pp. 16–34. Springer (2014)
36. Proper, H.A., Bjeković, M., Feltus, C., Razo-Zapata, I.S.: On the Development of a Modelling Framework for Value Co-creation. In: Proc. of the 12th Int. Workshop on Value Modelling and Business Ontologies (VMBO) (2018)
37. Proper, H.A., Bjeković, M., van Gils, B., Hoppenbrouwers, S.J.B.A.: Towards Grounded Enterprise Modelling. In: Proc. of the Fact-based Modelling Workshop (FBM 2017), held at the On The Move (OTM) conference. Springer (2017), post-proceedings, to appear
38. Proper, H.A., Lankhorst, M.M.: Enterprise Architecture – Towards essential sense-making. Enterprise Modelling and Information Systems Architectures (EMISA) **9**(1), 5–21 (June 2014)
39. Proper, H.A., Op 't Land, M.: Lines in the Water – The Line of Reasoning in an Enterprise Engineering Case Study from the Public Sector. In: Harmsen, A.F., Proper, H.A., Schalkwijk, F., Barjis, J., Overbeek, S.J. (eds.) Proc. of the 2nd Working Conf. on Practice-driven Research on Enterprise Transformation (PRET 2010). LNBIP, vol. 69, pp. 193–216. Springer, Delft, the Netherlands (November 2010)
40. Proper, H.A., Winter, R., Aier, S., de Kinderen, S. (eds.): Architectural Coordination of Enterprise Transformation. Enterprise Engineering Series, Springer (2018)
41. Razo-Zapata, I.S., Chew, E., Proper, H.A.: Visual Modeling for Value (Co-)Creation. In: Proc. of the 10th Int. Workshop on Value Modelling and Business Ontologies (VMBO) (2016)
42. Razo-Zapata, I.S., Chew, E., Proper, H.A.: Towards VIVA: A VIsual Language to Model VAlue Co-creation. In: The 5th Int. Conf. on Serviceology, (2017)
43. Razo-Zapata, I.S., Chew, E., Proper, H.A.: VIVA: A VIsual Language to Design VAlue Co-creation. In: The 20th IEEE Int. Conf. on Business Informatics (CBI 2018) (2018)
44. Rothrock, L., Yin, J.: Integrating Compensatory and Noncompensatory Decision-Making Strategies in Dynamic Task Environments. In: Kugler, T., Smith, J.C., Connolly, T., Son, Y.J. (eds.) Decision Modeling and Behavior in Complex and Uncertain Environments, Springer Optimization and Its Applications, vol. 21, pp. 125–141. Springer (2008)
45. Rouwette, E.A.J.A., Vennix, J.A.M.: System Dynamics and Organizational Interventions. Systems Research and Behavioral Science **23**, 451–466 (2006)
46. Sandkuhl, K., Stirna, J., Persson, A., Wißotzki, M.: Enterprise Modeling: Tackling Business Challenges with the 4EM Method. Springer (2014)
47. Snoeck, M.: Enterprise Information Systems Engineering – The MERODE Approach. Enterprise Engineering Series, Springer (2014)
48. The Open Group: TOGAF Version 9.1. Van Haren Publishing, 10th edn. (2011)
49. Tribolet, J., Sousa, P., Caetano, A.: The Role of Enterprise Governance and Cartography in Enterprise Engineering. Enterprise Modelling and Information Systems Architectures (EMISA) **9**(1), 38–49 (June 2014)
50. Tulinayo, F.P., van Bommel, P., Proper, H.A.: Enhancing the System Dynamics Modeling Proces with a Domain Modeling Method. Int. Journal of Cooperative Information Systems **22**(02) (2013)
51. Vargo, S.L., Lusch, R.F.: Service-dominant logic: continuing the evolution. Journal of the Academy of marketing Science **36**, 1–10 (2008)
52. Vargo, S.L., Lusch, R.F.: Institutions and axioms: an extension and update of service-dominant logic. Journal of the Academy of Marketing Science **44**(1) (January 2016)
53. Wout, J.v., Waage, M., Hartman, H., Stahlecker, M., Hofman, A.: The Integrated Architecture Framework Explained. Springer (2010)
54. Zachman, J.A.: A framework for information systems architecture. IBM Systems J., **26**(3), 276–292 (1987)