On the Development of Enterprise-Grade Tool Support for the DEMO Method

Mark A. T. Mulder[1,2 [0000−0002−1846−0238]] and
Hend*erik* A. Proper[3,4 [0000−0002−7318−2496]]

[1] TEEC2, Hoevelaken, the Netherlands
[2] Radboud University, Nijmegen, the Netherlands
[3] Luxembourg Institute of Science and Technology (LIST), Belval, Luxembourg
[4] University of Luxembourg, Luxembourg
markmulder@teec2.nl, e.proper@acm.org

Abstract. The Design and Engineering Methodology for Organisations (DEMO) is a core method within the discipline of Enterprise Engineering (EE). It enables the creation of so-called *essential* models of organisations. Such models are enterprise models that focus on the organisational essence of an organisation. They do so primarily in terms of the actor roles involved, and the business transactions between them. The DEMO method has a firm theoretical foundation. At the same time, there is increasing uptake of DEMO in practice. This uptake also results in a need for enterprise-grade tool support for the use of the method. In this experience paper, we report on experiences in the development of enterprise-grade tool support for the practical use of DEMO.

Keywords: enterprise engineering · DEMO · modelling tools

1 Introduction

This *experience paper* reports on experiences in the development of enterprise-grade tool support for the Design and Engineering Methodology for Organisations (DEMO) method. DEMO [6] is a key method within the discipline of Enterprise Engineering (EE) [7]. It focuses on the creation of *essential* models of organisations in terms of the actor roles involved, as well as the business transactions between these actor roles.

DEMO has strong methodological, and theoretical, roots [6, 7]. At the same time, there is increasing uptake of DEMO in practice. The latter is illustrated by the active usage (and certification) community[5], reported cases concerning the use of DEMO [2, 8], as well as the integration with other mainstream enterprise modelling approaches such as ArchiMate [15, 31] and BPMN [19, 12].

The increased uptake of DEMO has also triggered the need for enterprise-grade tool support. Due to its academic roots, there was not much attention for the development of enterprise-grade tool support (we will discuss the notion of enterprise-grade in more detail in section 2) so far. Tools supporting DEMO have indeed been developed. However,

[5] http://www.ee-institute.org/en
https://www.linkedin.com/company/enterprise-engineering-institute

these generally involve either (advanced) academic prototypes, or are provided by smaller "boutique" tool vendors. These tools, regretfully, do not classify as enterprise-grade tooling (see [23] for more details). Meanwhile, however, the lack of such tool support is now also seen as hampering the further uptake of DEMO by larger organisations.

The focus of this (empirical) paper is on our experience in the development of such enterprise-grade tool support for the DEMO method on top of the Sparx Enterprise Architect (SEA) platform. The development of tool support for DEMO also provided interesting insights into limitations of DEMO Specification Language (DEMOSL), the specification language that accompanies the DEMO method. These insights include: (1) the need to elaborate the meta-model of the method, e.g. enable verification and analysis, (2) the need to complement the core meta-model of the method with a visualisation oriented meta-model dealing, and (3) the need to be able to exchange the models between tools, resulting in an exchange oriented meta-model. Each of these experiences will be illustrated in terms of the needed landscape of meta-model(s), as well as some fragments of the actual meta-models.

The work, as reported on in this paper, is part of a larger design science effort [25] to evolve two artefacts: (1) DEMOSL and (2) its associated tool support, towards a more enterprise-grade level. Design science, according to Wieringa [34], is the design and investigation of artefacts within a context that are designed to interact with this context and should improve something in that context. A (design science) artefact is an object that solves a problem by interaction with the context of that artefact. In our case, the involved artefacts are the DEMOSL and the associated tool support.

The remainder of this paper is structured as follows. Section 2 provides background on what we mean by "enterprise-grade" tool support. In section 3, we then provide background to the DEMO method, also illustrating its theoretical foundations, and its uptake in practice. We then continue in section 4 with the need to extend, and even improve, the original meta-model to e.g. enable model verification. Section 5 then takes us to the need to be able to visualise models; both in order to create / edit models as well as to communicate them. Before concluding, section 6 discusses the consequences of the need to be able to exchange models between tools.

2 Enterprise-Grade Tool Support

The term "enterprise-grade" does not originate from science[6], but is a term commonly used in practice. As reported by practitioners, the term is used to differentiate consumer products from enterprise products [29]. In [10], Gartner defines it as: "*Enterprise-grade describes products that integrate into an infrastructure with a minimum of complexity and offer transparent proxy support.*" In line with this, enterprise-grade (in the context of software applications in general) is associated by practitioners [29, 32, 14] to characteristics such as *productivity*, *security* (including e.g. encryption of data, data security, granular levels of user access, protection of data, compliance), *integration*, *administration*, *support*, and *scalability*.

[6] Even though we can not claim to have conducted an in-depth literature survey on the term "enterprise-grade", the papers we did find (through a basic google scholar search) left the definition implicit.

For a modelling tool to be enterprise-grade, it also implies that there must be a solid and secure repository (to store the models). Essential qualities for such repositories include *extensibility*, *maintainability*, *interoperability*, and *portability* [26]. Since enterprise engineering and architecting efforts typically involve many different aspects, as well as many different stakeholders [28, 33], it is important for modelling tools to provide different visualisations. Even more, these visualisations should be in a format that can be easily integrated into standard presentation and text editing software.

The above considerations provided the motivations (as reported in [23]) to develop support for DEMO on top of the Sparx Enterprise Architect (SEA) platform. The SEA platform is widely used in the industry for enterprise modelling, supports multiple models, has a repository, can apply consistency rules to the stored models, and is extendable. In line with this, the remainder of the paper focuses on the experiences in realising DEMO support on top of the SEA platform.

3 DEMO Background

As mentioned before, DEMO [6] is a core method within the discipline of Enterprise Engineering (EE) [7]. The method has strong theoretical roots [30, 4, 5, 6, 7], and focuses on the creation of so-called *essential models* of organisations. Such essential models aim to capture the organisational essence of an organisation, primarily in terms of the (human) actor roles involved, the business transactions [30, 4] between these actor roles, as well as the processes needed in their realisation. From a theoretical perspective [30, 4], the business transactions are based on speech acts [13] between the actors who enact these roles. DEMO supports the modelling of the overall, as well as the more detailed, processes in an organisation, and the underlying information processing. Its strong focus on identifying the business transactions between (human) actor roles in order to capture the organisational essence, is what makes DEMO complementary to main stream enterprise modelling standards such as BPMN [27] and ArchiMate [17, 3]. This complementarity has also been the subject of publications reporting on the combination of DEMO and e.g. BPMN [19, 12], ArchiMate [31, 15], or e3Value [16].

DEMO is model-driven in the sense that models are used as its mainstream of deliverables. As such, it also fits well in the context of Model-Driven Engineering. More specifically, the earlier mentioned essential models comprise of four (integrated) aspect models: the Construction Model (CM), the Action Model (AM), the Process Model (PM) and the Fact Model (FM). Each of these models is expressed in one or more diagrams and one or more cross-model tables providing viewpoints on the complete model.

The aspect models overlap in elements; specifically, the concept of *transaction kind* appears in all aspect models; therefore, being the de-facto integration point of the model as a whole. A complete DEMO model covers the product, process, and information view on the essential organisations. These views are linked by the action rules that, in organisations, are often called business rules.

DEMO is built on a set of theories, where the Performance in Social Interaction theory (PSI-theory) constitute the foundation of Enterprise Ontology (EO) from a construction perspective. The PSI-theory divides an organisation into three worlds of acts and facts that reach commitments and agreements in predictable phases. One composes these agreements in a logical order that demands the right responsibilities of the right subjects. Finally, the subjects' capabilities are appointed into three categories to make distinctions in decisions and other actions. This standard pattern and production can be used anywhere in the organisation to enhance communication between subjects.

The PSI-theory divides the world into three worlds: actors and actor roles (a-world), communication (c-world) and products (p-world). Each of the c- and p-worlds may perform acts and produce facts. The communication acts, performed by actor-roles, produce communication facts. The same rule holds for the p-world. In short, actors perform communication and production acts, thereby creating communication facts about the creation of production facts. Furthermore, the PSI-theory defines the communication pattern between two actor roles. Every commitment to reach a production result follows this universal pattern, again expressed in the transaction kind.

Construction Model – The CM is the first and most comprehensive model to produce when modelling an organisation in DEMO, applying the Organisational Essence Revealing (OER) method. A CM is a model that represents the construction of an organisation. This model consists of the identified transaction kinds and the actor roles that are either executor or initiator of these transaction kinds. The resulting 'network' of transaction kinds and actor roles is always a set of tree structures, which arise from the inherent property that every transaction kind has exactly one elementary actor role as its executor (and vice versa) and that every actor role may be the initiator of no, one or more transaction kinds. The Construction Model (CM) involves the Organisation Construction Diagram (OCD) showing the Transaction Kind (TK), Aggregate Transaction Kind (ATK), Elementary Actor Role (EAR), Composite Actor Role (CAR) within a Scope of Interest (SoI). These diagrams show the dependencies between roles in execution and information. The high abstraction level makes this a compact diagram in relation to the implementation of the organisation.

The Transaction Product Table (TPT) shows the TK identification and description together with the product identification and description. This table is used to get insight into the products that are being created in the organisation.

Finally, the Bank Contents Table (BCT) shows the contents of the ATK. This contains the identification and name of the ATK and the Entity Type (ET) and attributes of those ETs that are present. This is used to show the extend of (external) data.

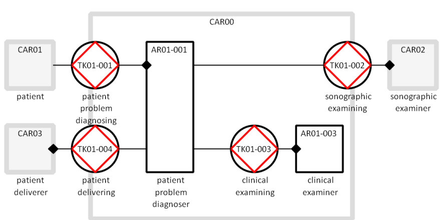

Fig. 1. Poligyn OCD modelled in the tool

Name	Alias	Product Kind Formulation
TK01-004	patient delivering	the patient of [patient problem] is delivered
TK01-003	clinical examining	[clinical examination] is performed
TK01-002	sonographic examining	[sonographic examination] is completed
TK01-001	patient problem diagnosing	[patient problem] is diagnosed

Fig. 2. Poligyn TPT modelled in the tool

Process Model – The PM bridges its CM and the coordination part of its AM. To this end, it specifies how the transaction kinds in a tree are related to each other. More precisely, it specifies which transaction steps in an enclosed transaction kind are connected to which steps in the enclosing transaction kind, and by which kind of link (response-link or waitlink). The PM involves a single diagram kind, the Process Structure Diagram (PSD), which shows the relations between the process steps of interrelated transactions. This is used to explain the order and dependencies between transactions. Business rules are partially covered as well.

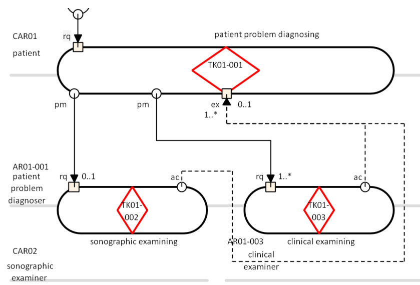

Fig. 3. Poligyn PSD modelled in the tool

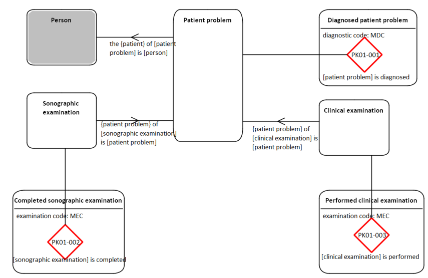

Fig. 4. Poligyn OFD modelled in the tool

Fact Model – The FM of a Scope of Interest bridges its CM and the production part of its AM. It specifies the various entity types, property types, attribute types, and entity types, as well as their mutual relationships. The FM also involves a single diagram kind, the Object Fact Diagram (OFD), which shows ETs and Product Kinds (PKs), and the Information Use Table (IUT). This model is often called the data model, although it shows much more information.

Action Model – The AM comprises the guidelines that guide actors in doing their work, i.e. performing their coordination acts and their production acts. Action rules, which are actually (imperative) business rules, guide actors in responding to the coordination events they have to deal with. A semi-structured English-like language is used to express the action rules. Work instructions guide actors in performing the production acts, i.e. in bringing about the products of transactions.

This last aspect model contains Action Rules Specification (ARS) and Work Instruction Specification (WIS). Per non-trivial process step minimal one specification shows the input and conditions to proceed in the transaction pattern or advance to other transactions. This specification is used to model all details of the (to-be) organisation.

Different aspect models contain overlapping elements; therefore, the DEMO *essential model* is the result of the combination of all aspect models.

We modelled the example case of [8, chapter 17], named Poligyn using the tool. In fig. 1 we see the four TK of the CM with its initiating and executing actor roles within the modelled organisation. These TKs can also be shown in a table as the TPT in fig. 2. One step further in the analysis, we get to the process dependencies depicted in fig. 3. Finally, in fig. 4 the FM shows the internal references between ETs and the relations between ET and TK.

As discussed in the introduction, there has been an increasing uptake in practice of DEMO. This can be partially attributed to the complementary perspective that DEMO offers next to mainstream enterprise modelling languages such as ArchiMate and BPMN. Meanwhile, DEMO has a proven track record in process (re)design and reorganisation, software specifications based on the organisation, modelling business rules and proving General Data Protection Regulation (GDPR) [11] and other business Information Technology Infrastructure Library (ITIL), Business information Services Library (BiSL), and NEderlandse Norm (NEN) norm compliance [24]. Additionally, an early descendant of DEMO, called 'Voorwaarden scheppen voor de Invoering van Standaardisatie ICT in de bouw' (VISI), has evolved into the ISO 29481-2:2012[7] (BIM related) standard for the construction sector.

4 Meta-Model Extensions

A first issue we encountered in the development of tool support for DEMO was that, even though it has a thorough theoretical basis, the original meta-model of DEMO was not specific and detailed enough to enable an immediate implementation. The latter can be explained by the fact that the book defining the DEMO method [6] was primarily written to teach learners (students and practitioners) to create DEMO models in accordance with the DEMO way of thinking, and draw (human to human) communicable DEMO models to reason about the organisation. As such, the book puts the priority on "doing", when introducing the different DEMO aspect model kinds. There was no need for a strict meta-model. In line with this, the formalisation(s) provided in the original DEMO book [6] aimed to support didactic goals rather than the development of automated modelling tools. As such, it was never meant to provide formal meta-models that would enable the development of, and automated support for, the methodology.

While a good theory should help professionals to do their job, more professionals are raised using automation as their main tooling. Where pencils were used, tablets have taken their place, and people see automation as their starting point. Tooling fulfils a lever function in broadening the use of the methodology. As a result, to enable tool development, a more complete and detailed formalisation and meta-model were needed. This resulted in the development, and evolution, of the DEMOSL [9], the specification language for DEMO. A first validation of the meta-model of this specification language was reported in [21]. As part of the (partial) validation [21], all existing DEMO (example) aspect models (taken from the official course material and practical cases) were positioned within the specification language to see if they fitted.

Beyond the ability to "capture" actual DEMO models in the specification language, more extensions of the meta-model were needed to enable verification of the models, and the operational use of the method in practice [22]. In the remainder of this section, we discuss some examples of changes / refinements that were made in relation to the original meta-model, the results of which are summarised in fig. 6.

For instance, when modelling an organisation, the functional components like departments are needed for stakeholders to understand the structures of the organisation. Given DEMO's focus on the ontological essence of organisations, it does not provide functional concepts like departments. It does, however, provide the constructional concept of Composite Actor Role (CAR; see the previous section). CARs are now allowed to be used to represent functional concepts such as departments, enabling modellers to model departments (or other organisational units) as organisations within organisations. This hierarchical relation has been added to the meta-model.

[7] `https://www.iso.org/obp/ui/#iso:std:iso:29481:-2:ed-1:v1:en`

It also turned out that, for practical purposes, some concepts could actually be removed from the meta-model. For instance, according to the theory, each Organisation Construction Diagram (OCD) corresponds to an explicitly defined Scope of Interest (SoI). In practice, however, the SoI always corresponds directly to a Composite Actor Role (CAR). As a consequence, in practice, the SoI concept is redundant, and, therefore, does not have to be included in the meta-model explicitly.

A further issue pertained to the fact that the DEMO method allows modellers to start from any of the four aspect models. When learning the method, one generally starts with a Construction Model (CM) and gradually works down to the Action Model (AM; see fig. 5). However, in practice, when interviewing domain experts in an organisation, these experts usually talk about the existing process and associated rules. In other words, starting with AM related information first. Additionally, the Fact Model (FM) information about entity types and attribute type is also provided relatively early when interviewing the domain experts.

The original DEMO meta-model did not allow for models to "grow" from the different aspect models, in the sense that the consistency rules would require the model to always be complete as a whole (so, including all aspect models). Therefore adjustments to the meta-model needed to be made to allow for such flexibility, while still enforcing (at the end of the modelling process) the overall consistency.

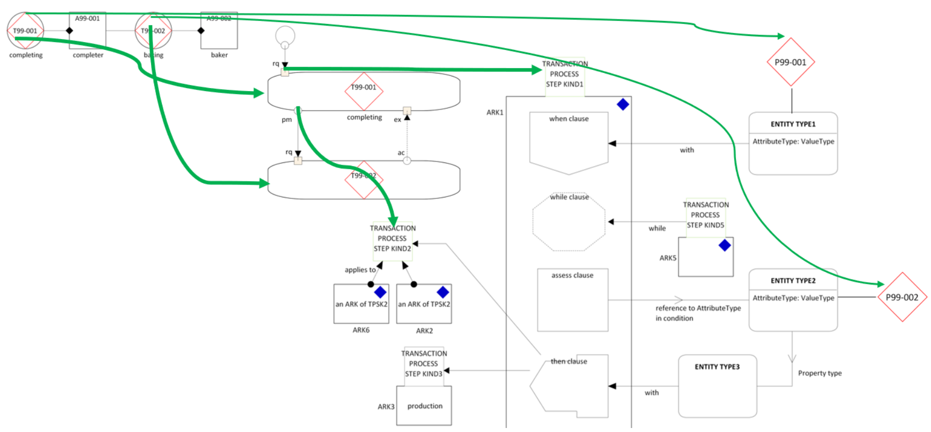

Fig. 5. CM-AM flow

In practice, organisation models that resulted from the OER analysis often raised questions regarding the *origins* of the included transaction kinds. More specifically, a cross-reference from the elements in the organisation model to the original OER analysis was missing. To support this kind of cross-reference, we have added the interview and interview-line concept into the meta-model. By registering every aspect of the OER analysis as a connection from the interview-line to the elements modelled from that line information we have created a cross-reference from the source to the final model.

Another finding is that in practice, there was a need for DEMO models to be related to their existing / planned implementation. For instance, a "serving" connector was introduced that can be used to connect e.g. DEMO model elements to e.g. application components in the ArchiMate's Application layer. With this connector, one can, for instance, point to application components that implement the transaction kind or Transaction Process Step Kind (TPSK). Another example of the need to be able to include more of the implementation context involves the introduction of an actor (type) that aggregate actor roles. Such actors types correspond to job functions and, therefore, combine all competences of the actor roles that are aggregated. This can be used for HR implementation information.

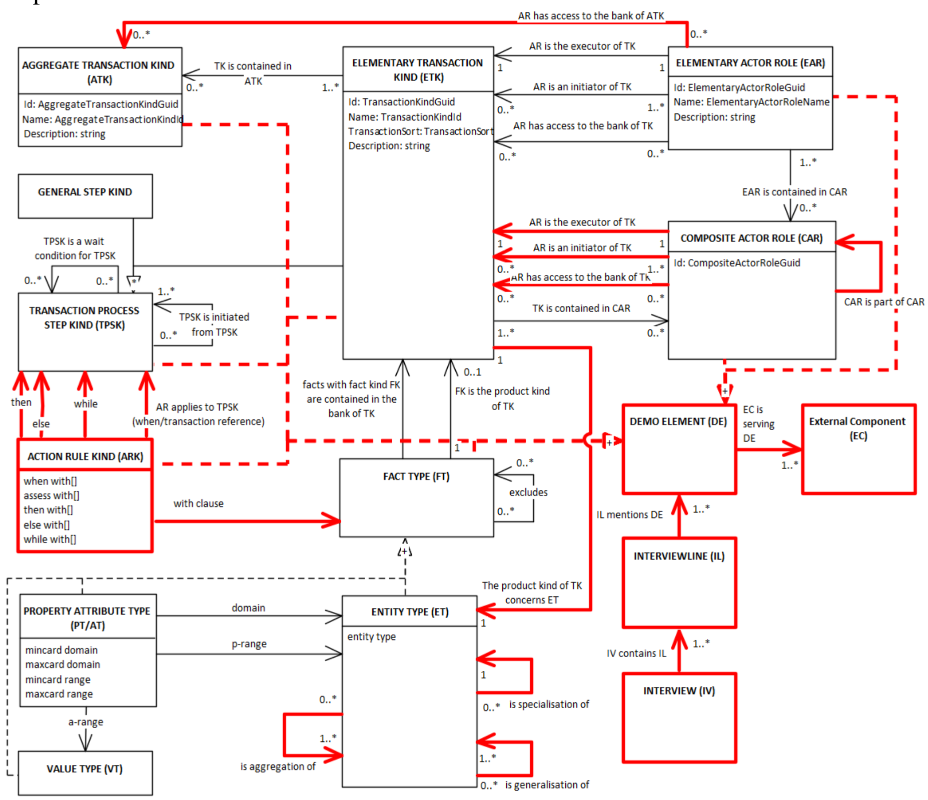

Fig. 6. Extension of DEMO meta-model (red bold lines are new)

Finally, the action rule specification, as described in the DEMO method and associated specification language, consists of a semi formalised way to describe the communication actions that result in a decision on either actor role. To start using the Action Rules Specification (ARS) in a more formal way, a more elaborate definition was needed. Therefore, based on earlier work [1], a grammar to represent action rules was created.

Figure 6 summarises the (discussed) extensions of DEMOs original meta-model, while fig. 7 shows both the tool “in action”, as well as some details of the tool’s *implementation* on the SEA platform. In the latter, we see (numbers referring to the respective windows as displayed in the figure) how SEA provides a set of base classes (1) for elements, connectors, diagrams, and toolboxes. Each DEMO model element involves an extension of one of the base classes of SEA (2), while also having custom properties based on the DEMO meta-model (3). To enable different visualisations, SEA provides simple scripts that enable the drawing of shapes (4). SEA provides interface

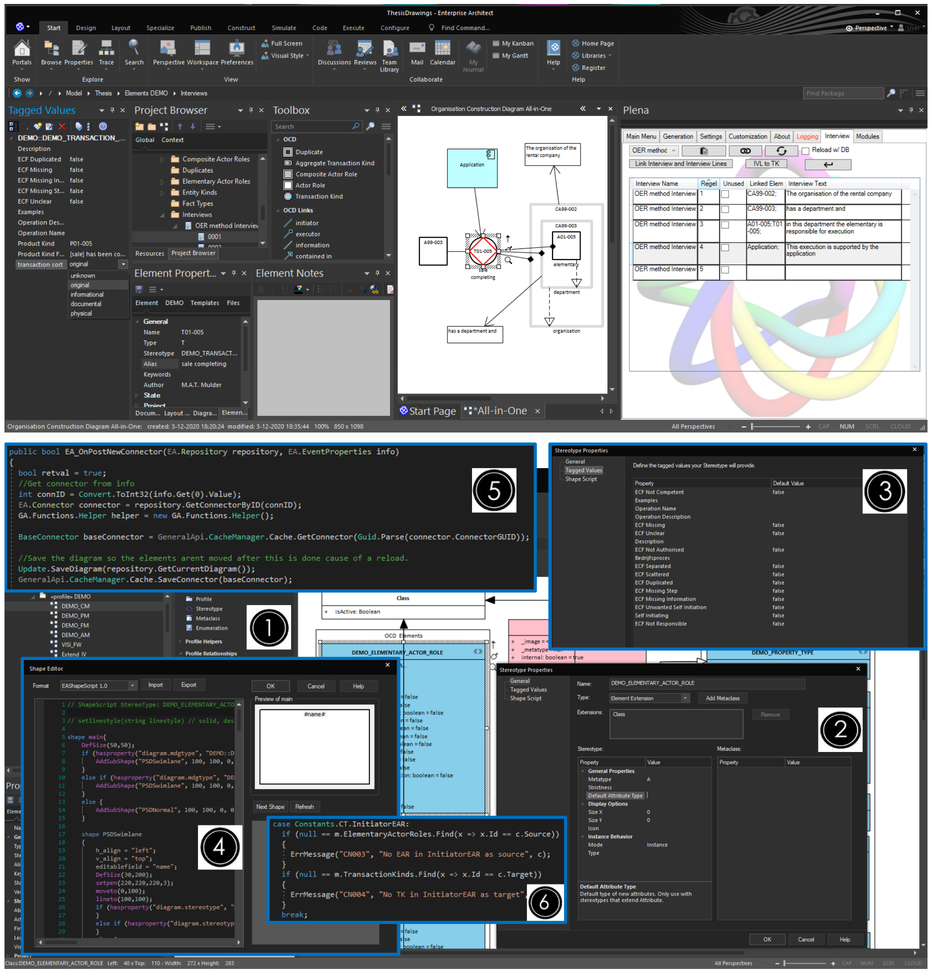

Fig. 7. Tool in action, and its implementation

"hooks", that are triggered when editing models in the graphical user interface, that can be used by add-on applications to provide functionality that is not available in SEA (5). The verification of DEMO models has been implemented in C# using these "hooks" (6).

The meta-models have been updated according to fig. 6. These models are created for tool developers and the users have no meddling in these meta-models. The models improve the usability of the methodology through tool usage and the methodology, therefore, becomes even more powerful in performance than without tooling.

5 Visualisation

Since the DEMOSL [9] primarily focuses on consistency and completeness of DEMO models from a "content" perspective, it does not include any specifics about the actual representation of these models in terms of diagrams, tables and other possible visualisations. As such, the DEMOSL does not provide guidelines regarding the concrete syntax of models in terms of e.g. shapes and icons to be used. When examining a corpus of models produced across different cases, we found various variations in drawings of elements that each could be interpreted as the convention of those elements. In moving towards (standardised) tool support, this had to be remedied in terms of an explicit meta-model of the allowed visualisations.

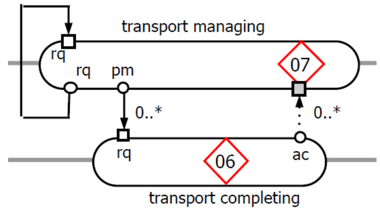

Fig. 8. PSD specification example

As an example, consider the diagram provided in fig. 8. In an OCD, the shape is normally drawn as a circle enclosing a diamond and this circle is stretched in the PSD with the diamond displayed at a seemingly random position within this "stretched circle". In fig. 8, the name of the transaction appears below or above or at the side-top of the transaction fig. 8, the diamond is at a certain percentage from the left side, the stretching is a random length, while the swim lane usage is not consistent (i.e. 07 has the same initiator and executor but is visualised on top of two swim lanes).

Furthermore, the examples of the same case are different between different DEMO versions without explaining why this is the case, making it impossible to give guidance to readers that were trained in a different version to interpret the visualisation, and also making it impossible for a tool to comply to the specifications.

```
sincerity: <no specific condition>
truth:
  for each Order Item in the contents of
      Order:
    the pizza kind of Order Item is
        available

       the amount of Order Item is
           producible
    the requested production time of
        Order is greater than Now plus 10
        minutes
```

```
truth:
  for each Order Item in {the contents of
       Order} {
    the availability of pizza kind of
        Order Item is equal to
        true;
    the [producebility of amount] of
        Order Item is equal to true;
    the requested production time of
        Order is greater than the [Now
        plus10 minutes] of Time;
  }
```

Fig. 9. ARS grammar upgrade example

Due to the complexity reduction of the CM the resulting aspect model visualisation is not readable for stakeholders that expect a flow. This flow is only available when modellers have finished the analysis from CM through PM to AM. The DEMO specification has a language for the AM that is semi-formalised and we tried to create a formalised form for that model (see fig. 9). This specification, though formally more correct, is still not readable for many stakeholders. Visualising the final aspect model AM in a recognisable form has positive effects on the comprehension with stakeholders of the modelling of DEMO.

When visualising an (aspect) model the visualisation is a representation of the model. On a meta-level one needs to define the elements that are visualised and the components of those elements of the meta-model that are used in this visualisation. The omission of this component visualisation definition makes the visualisation arbitrary. Defining these visualisations requires a mapping specification of every visualisation to the entity and attribute of the meta-model it is representing. These mappings also need translations from available values to required visualisations. For example, the transaction sort of a Elementary Transaction Kind (ETK) is defined as original, informational or documental and can be visualised as a red, green or blue diamond, respectively. While creating the models and tooling and trying to visualise the model for various stakeholders this omission of the mapping became apparent, whereas this never happens with the toy examples from the educational material.

The mapping of the visualisation is the definition of viewpoints of DEMO modelling. At this moment just a few viewpoints have been defined in table 1, but more exist and can be defined for specific stakeholders. Viewpoints that cover multiple notations have to be defined to cover the representation combinations of DEMO and other notations (e.g. ArchiMate).

Each viewpoint needs, next to the definition and mapping, the semantic explanation of the information represented. Not only does every element and connection needs an explanation, also the whole viewpoint needs an explanation, and even a definition. The current viewpoints (e.g. diagrams and tables) do not have a definition or explanation of what they represent. The OFD does show the entity types, attribute types, property types and product types (the product type is another viewpoint of the ETK) but it is not defined what is represented. It has been interpreted as the data model, the creation of data due to transaction acts, the relational model, the possible-state model and the authorisation model. Each interpretation of the OFD is possible and, therefore, it is likely that the OFD contains too much information. Our experience tells us that the richness of each of the DEMO diagrams and tables is too much for the average stakeholder to grasp.

Diagram → Entity Types ↓	OCD	PSD	TPD	OFD	ARD	RHD	AFD	..
ETK	X	X	X	X	–	–	–	–
ATK	X	–	–	–	–	–	–	–
EAR	X	X	–	–	–	X	X	–
CAR	X	X	–	–	–	X	–	–
ET	–	–	–	X	–	–	–	–
CET	–	–	–	X	–	–	–	–
TPSK	–	X	X	–	X	–	–	–
ARS	–	–	–	–	X	–	–	–

Table 1. Elements and Diagrams

At the moment of writing this paper, a list of viewpoints of a DEMO model exists in the tool. For example (see table 1): OCD, PSD, Transaction Pattern Diagram (TPD), OFD, Action Rules Diagram (ARD), Role Hierarchy Diagram (RHD), and Actor role Function Diagram (AFD) These diagrams contain various elements of the DEMO model: ETK, ATK, EAR, CAR, ET, Composite Entity Type (CET), TPSK, and ARS.

As mentioned before in our discussion regarding the meaning of enterprise-grade in section 2, since enterprise engineering typically involves many different aspects, as well as many different stakeholders [28, 18, 33], it is important for modelling tools to provide different visualisations. For instance, the OCD does not work for C-level stakeholders [22] C-level prefers high-level blocks and arrow representations of processes, such as provided by the ARS level diagrams. Similarly, towards C-level management, the PSD also do not work well when dealing with complex processes. Finally, the OFD, when completely drawn, is also uneasy to read for a lot of people. Dividing it into several partial models helps to make partial models clear to stakeholders.

As such, in line with [18], it is necessary to distinguish between different visualisation strategies for different stakeholder groups and different purposes. For instance, visualisation can also involve the creation of animation of a process, or even gamification. Enabling stakeholders to simulate, or even "play", their processes allows them to see the modelled processes as they are running. It also combines the view on processes with the production facts that are created in a transaction.

Our first attempt with gamification of DEMO models (see fig. 10) visualises the OCD in a 3D setting where the actor roles are visualised as people doing tasks behind their desks. This visualisation is supported by a multi-user environment that can play one or more roles in the gamification scope. The setting allows for multiple roles played by a single player, therefore, restricting the execution of multiple roles with only one actor to fulfil all labour.

Fig. 10. Simulation of a DEMO model

These experiences require new visualisations to be created, evaluated and improved. When new insights appear, first, the meta-model needs to be extended with all new concepts that emerge in the visualisation and implementation activities. Furthermore, viewpoints can be defined and reference the existing and new concepts in the meta-model. Next, new diagrams and toolboxes with elements and connectors can be added to the tool where the basics functionality of SEA allows for. Finally, the functionality that cannot be solved by configuration will be implemented in the add-on.

6 Model Exchange

As reported in earlier work [23], a number of tools[8] can model (partial) DEMO models, while some of these even have a build-in engine to execute the actual models. It would be beneficial to be able to export and import DEMO models across tools because specific tools can have specific benefits when used in a certain organisational environment.

In addition, given the complementarity between e.g. BPMN, ArchiMate, e3Value, and DEMO, it would be beneficial to also bridge between the respective models. Experimental results regarding the potential benefits of this have been reported in

[8] Created by e.g. Bakker&Spees, Technia, Future Insight, and Formetis.

e.g. [19, 12, 31, 15, 16]. This requires DEMO modelling tools to be able to export (and import) DEMO models. To this end, an exchange meta-model has been created that enables the exchange of both the actual model, including the different aspect models, as well as their visualisations. The exchange meta-model enables translation and connection to various other languages and includes model extensions to store models in other modelling languages such as ArchiMate and Business Process Model and Notation (BPMN). The exchange meta-model is based on XML Schema Definition (XSD), due to its common availability in modern-day development environments [20].

The SEA based modelling tool has been extended with this export capability. Both as a way to validate the exchange meta-model, and to experiment with the export of DEMO models towards a gamification application (as illustrated in fig. 10).

A specific class of model exchanges needed in practice is the exchange between DEMO and VISI. As mentioned in section 3, VISI is an early descendant of DEMO which has now evolved into the ISO 29481-2:2012[7] (BIM-related) standard for the construction sector. The exchange model between DEMO and VISI has been defined in the standard ISO 29481-2:2012.

7 Conclusions and Future Research

In this experience paper, we reported on some of the experiences in the development of enterprise-grade tool support for the practical use of DEMO. The chosen platform to realise this was the SEA platform. Some of the lessons learned from this research effort, that might be applicable to the development of modelling tools for other modelling languages as well, are:

1. There is a considerable difference between the meta-model used for educational (and theory development) purposes compared to the meta-model needed for actual tool development. Especially when also taking the experiences from practitioners using the modelling language in real-world projects into account
2. The core meta-model of the modelling language also needed to be extended with a visualisation meta-model to accommodate for different visualisations (of the same models).
3. Usage of a generic enterprise-grade tool platform, such as SEA provides a quick path towards enterprise-grade tool support. From a development perspective, the development of a bespoke solution may provide more flexibility. However, the usage of an existing tool (enterprise-grade) platform increases the initial acceptance.

In future research, we expect more feedback from practitioners based on the day-to-day use of DEMO. We also expect this to result in the need for new visualisations (including simulation and gamification) catering to the needs of different stakeholders. New research efforts have already been started to get more insight into the relation between the complexity of the information and the way we try to communicate this information.

In addition, we expect that further extensions to the core meta-model will be needed to enable sensible connections to complementary modelling languages and methods.

References

1. Andrade, M., Aveiro, D., Pinto, D.: Bridging ontology and implementation with a new demo action meta-model and engine. In: Enterprise Engineering Working Conference. pp. 66–82. Springer (2019)
2. Aveiro, D., D. Pinto, D.: A case study based new DEMO way of working and collaborative tooling. In: 2013 IEEE 15th Conference on Business Informatics. pp. 21–26. IEEE Computer Society Press, Los Alamitos, California (2013)
3. 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)
4. Dietz, J.L.G.: Generic Recurrent Patterns in Business Processes. In: Aalst, W.M.P.v.d., Hofstede, A.H.M.t., Weske, M. (eds.) Proceedings of the International Conference on Business Process Management (BPM2003), Eindhoven, the Netherlands. LNCS, vol. 2678, pp. 1018–1033. Springer (June 2003)
5. Dietz, J.L.G.: A World Ontology Specification Language. In: Meersman, R., Tari, Z., Herrero, P. (eds.) On the Move to Meaningful Internet Systems 2005: OTM 2005 Workshops, OTM Confederated International Workshops and Posters, AWeSOMe, CAMS, GADA, MIOS+INTEROP, ORM, PhDS, SeBGIS, SWWS, and WOSE 2005, Agia Napa, Cyprus, October 31 - November 4, 2005, Proceedings. LNCS, vol. 3762, pp. 688–699. Springer (October/November 2005)
6. Dietz, J.L.G.: Enterprise Ontology – Theory and Methodology. Springer (2006)
7. Dietz, J.L.G., Hoogervorst, J.A.P., Albani, A., Aveiro, D., Babkin, E., Barjis, J., Caetano, A., Huysmans, P., Iijima, J., Kervel, S.J.H.v., Mulder, H., Op 't Land, M., Proper, H.A., Sanz, J., Terlouw, L., Tribolet, J.M., Verelst, J., Winter, R.: The discipline of enterprise engineering. International Journal Organisational Design and Engineering **3**(1), 86–114 (2013)
8. Dietz, J.L.G., Mulder, J.B.F.: Enterprise Ontology – A Human-Centric Approach to Understanding the Essence of Organisation. The Enterprise Engineering Series, Springer (2020)
9. Dietz, J.L.G., Mulder, M.A.T.: Demo specification language 3.7 (2017), `https://www.eei-test.nl/mdocs-posts/demo-specification-language-3-7/`
10. Gartner: Information technology glossary. website (Nov 2020), `https://www.gartner.com/en/information-technology/glossary/enterprise-grade`
11. Gouveia, D., Aveiro, D.: Modeling the system described by the eu general data protection regulation with demo. In: Enterprise Engineering Working Conference. pp. 144–158. Springer (2018)
12. Gray, T., Bork, D., De Vries, M.: A new DEMO modelling tool that facilitates model transformations. In: Enterprise, Business-Process and Information Systems Modeling, pp. 359–374. Springer (2020)
13. Habermas, J.: The Theory for Communicative Action: Reason and Rationalization of Society, vol. 1. Boston Beacon Press, Boston, Massachusetts (1984)
14. Kepes, B.: What does enterprise gradereally mean? website, `https://www.forbes.com/sites/benkepes/2013/12/18/what-does-enterprise-grade-really-mean`
15. de Kinderen, S., Gaaloul, K., Proper, H.A.: On transforming DEMO models to ArchiMate. In: Bider, I., Halpin, T.A., Krogstie, J., Nurcan, S., Proper, H.A., Schmidt, R., Soffer, P., Wrycza, S. (eds.) Enterprise, Business-Process and Information Systems Modeling - 13th International Conference, BPMDS 2012, 17th International Conference, EMMSAD 2012, and 5th EuroSymposium, held at CAiSE 2012, Gdańsk, Poland, June 25-26, 2012. Proceedings. LNBIP, vol. 113, pp. 270–284. Springer (June 2012)
16. de Kinderen, S., Gaaloul, K., Proper, H.A.: Bridging value modelling to ArchiMate via transaction modelling. Software & Systems Modeling **13**(3), 1043–1057 (2014)

17. Lankhorst, M.M., Hoppenbrouwers, S.J.B.A., Jonkers, H., Proper, H.A., Torre, L.v.d., Arbab, F., Boer, F.S.d., Bonsangue, M.M., Iacob, M.E., Stam, A.W., Groenewegen, L., Buuren, R.v., Slagter, R.J., Campschroer, J., Steen, M.W.A., Bekius, S.F., Bosma, H., Cuvelier, M.J., ter Doest, H.W.L., van Eck, P.A.T., Fennema, P., Jacob, J., Janssen, W.P.M., Jonkers, H., Krukkert, D., van Leeuwen, D., Penders, P.G.M., Veldhuijzen van Zanten, G., Wieringa, R.J.: Enterprise Architecture at Work – Modelling, Communication and Analysis. Springer (2005)
18. Lankhorst, M.M., Torre, L.v.d., Proper, H.A., Arbab, F., Steen, M.W.A.: Viewpoints and visualisation. In: Enterprise Architecture at Work – Modelling, Communication and Analysis, pp. 171–214. The Enterprise Engineering Series, Springer, 4th edn. (2017)
19. Mráz, O., Náplava, P., Pergl, R., Skotnica, M.: Converting demo psi transaction pattern into bpmn: A complete method. In: Aveiro, D., Pergl, R., Guizzardi, G., Almeida, J.P., Magalhães, R., Lekkerkerk, H. (eds.) Advances in Enterprise Engineering XI. pp. 85–98. Springer International Publishing, Cham (2017)
20. Mulder, M.A.T.: Towards a complete metamodel for demo cm. In: OTM Confederated International Conferences" On the Move to Meaningful Internet Systems". pp. 97–106. Springer (2018)
21. Mulder, M.A.T.: Validating the demo specification language. In: Enterprise Engineering Working Conference. pp. 131–143. Springer (2018)
22. Mulder, M.A.T.: A design evaluation of an extension to the DEMO methodology, pp. 55–65. Advances in Enterprise Engineering XIII, Springer (2019)
23. Mulder, M.A.T., Proper, H.: Towards Enterprise-Grade Tool Support for DEMO. In: Grabis, J., Bork, D. (eds.) Practice of Enterprise Modelling (PoEM) 2020. LNBIP, vol. 400, pp. 90–105. Springer, Riga, Latvia (November 2020)
24. Mulder, M.A.T.: Nen 75xx modelled in demo. Unpublished manuscript (2020)
25. Mulder, M.: Enabling the automatic verification and exchange of DEMO models. Ph.D. thesis, Radboud University, Nijmegen, the Netherlands (Forthcomming)
26. Nunn, R.: What is "enterprise grade software"? website (Nov 2015), `http://tractsystems.com/what-is-enterprise-grade-software/`
27. OMG: Business Process Modeling Notation, V2.0. Tech. Rep. OMG Document Number: formal/2011-01-03, Object Management Group, Needham, Massachusetts (January 2011)
28. Op 't Land, M., Proper, H.A., Waage, M., Cloo, J., Steghuis, C.: Enterprise Architecture - Creating Value by Informed Governance. The Enterprise Engineering Series, Springer (2008)
29. Partridge, T.: What exactly does "enterprise-grade" mean?, `https://www.linkedin.com/pulse/what-exactly-does-enterprise-grade-mean-todd-partridge`
30. van Reijswoud, V.E., Mulder, J.B.F., Dietz, J.L.G.: Communicative Action Based Business Process and Information Modelling with DEMO. The Information Systems Journal **9**(2), 117–138 (1999)
31. Roland, E., Dietz, J.L.G.: ArchiMate and DEMO – Mates to Date? In: Albani, A., Barjis, J., Dietz, J.L.G. (eds.) Advances in Enterprise Engineering III, LNBIP, vol. 34, pp. 172–186. Springer (2009)
32. Sparks, G.: What does enterprise grade mean? website (Aug 2020), `https://www.quora.com/What-does-Enterprise-Grade-mean`
33. Wagter, R., Proper, H.A.: Involving the right stakeholders – Enterprise coherence governance. In: Proper, H.A., Winter, R., Aier, S., de Kinderen, S. (eds.) Architectural Coordination of Enterprise Transformation, chap. 10, pp. 99–110. The Enterprise Engineering Series, Springer (2018)
34. Wieringa, R.J.: Design science methodology for information systems and software engineering. Springer (2014)