A New DEMO Modelling Tool that Facilitates Model Transformations

Thomas Gray[1], Dominik Bork[2], and Marné De Vries[1(✉)]

[1] Department of Industrial and Systems Engineering, University of Pretoria, Pretoria, South Africa
{Thomas.Gray,Marne.DeVries}@up.ac.za
[2] Faculty of Computer Science, University of Vienna, Vienna, Austria
Dominik.Bork@univie.ac.at

Abstract. The age of digitization requires rapid design and re-design of enterprises. Rapid changes can be realized using conceptual modelling. The *design and engineering methodology for organizations* (DEMO) is an established modelling method for representing the organization domain of an enterprise. However, heterogeneity in enterprise design stakeholders generally demand for transformations between conceptual modelling languages. Specifically, in the case of DEMO, a transformation into business process modelling and notation (BPMN) models is desirable to account to both, the semantic sound foundation of the DEMO models, and the wide adoption of the de-facto industry standard BPMN. Model transformation can only be efficiently applied if tool support is available. Our research starts with a state-of-the-art analysis, comparing existing DEMO modelling tools. Using a design science research approach, our main contribution is the development of a DEMO modelling tool on the ADOxx platform. One of the main features of our tool is that it addresses stakeholder heterogeneity by enabling transformation of a DEMO organization construction diagram (OCD) into a BPMN collaboration diagram. A demonstration case shows the feasibility of our newly developed tool.

Keywords: DEMO · BPMN · ADOxx · Model transformation · Model consistency · Modelling tool

1 Introduction

The age of digitization requires rapid design and re-design of enterprises. In addition, the agile design paradigm embraces the use of multiple modelling languages to represent design knowledge. Unfortunately, this paradigm also has challenges regarding inconsistencies between model types that represent knowledge from the same knowledge domain. Modelling researchers should ensure to create models, languages, and methods that can be adapted to changing requirements in the future [1, p. 3].

Domain-specific languages are created to provide insight and understanding within a particular domain context and stakeholder group [2]. As an example, the *design and engineering methodology for organizations* (DEMO) provides models that represent the organization domain of an enterprise [3]. DEMO offers a unique design perspective, since

S. Nurcan et al. (Eds.): BPMDS 2020/EMMSAD 2020, LNBIP 387, pp. 359–374, 2020.
https://doi.org/10.1007/978-3-030-49418-6_25

its *four aspect models* have the ability to represent *organization design domain* knowledge in a concise and consistent way, removing technological realization and implementation details [3]. One of DEMO's aspect models, the construction model, incorporates an *organization construction diagram* (OCD) that provides a concise representation of enterprise operations. Managers value the OCD, since it becomes a blueprint that enables discussions on enterprise (re-)design and strategic alignment [3, 4]. Recker et al. [5] and Van Nuffel et al. [6] indicated that unguided use of the *Business Process Modeling Notation* (BPMN) constructs often leads to inconsistent models. It is thus our goal to combine the strengths of DEMO and BPMN by proposing a model transformation and modelling tool support.

Due to its characteristics of being consistent and concise, various authors experimented with transformations between modelling languages, as discussed in the remaining paragraph. De Kinderen, Gaaloul and Proper [7] indicated that "ArchiMate lacks specificity on how to model different perspectives *in-depth*" while [8, 9] add that ArchiMate lacks in expressing value exchange. As a solution to these deficiencies, [7] conducted a study to map concepts from DEMO to concepts contained within the business layer of the ArchiMate meta-model with the purpose of modelling the essential aspects of an enterprise first in DEMO, followed by a transformation into an ArchiMate model, adding technological realization and implementation details. Based on the work of Ceatano et al. [10] and Heller [11], Mraz et al. [12] presented transformation specifications to generate BPMN models from DEMO models. Yet, the specifications did not consider the complexity of hierarchical structures in DEMO models. In addition, their transformation specifications were not supported by tooling to automate DEMO-BPMN transformations.

This study starts with an evaluation of existing DEMO modelling tools. We conclude that existing modelling tools do not support all of DEMO's four *aspect models*. In addition, the tools do not facilitate transformations to other languages, such as BPMN. The main objective of this article is to address stakeholder heterogeneity by developing a DEMO modelling tool on the ADOxx platform. We demonstrate one of the main features of our tool, namely to transform a DEMO organization construction diagram (OCD) into a corresponding BPMN collaboration diagram.

The article is structured as follows. Section 2 provides background on multi-view modelling, as well as the existing knowledge on DEMO concepts that are explained via a demonstration case. Using design science research, as presented in Sect. 3, we present the requirements for a new DEMO tool in Sect. 4 and the DEMO constructional components that form part of the OMiLAB ecosystem, in Sect. 5. We also demonstrate the key functionality of the new DEMO tool, i.e. semi-automatic OCD-BPMN transformations for one out of four identified transformation scenarios. Section 6 ends with conclusions and suggestions for future research.

2 Background

Model-based development (MBD) approaches suggest separation of concerns, using multiple views, to manage the complexity of modern software systems [13]. Yet, one of the challenges of multi-view modelling is the lack of consistency management [14].

Bork [15] emphasised the need to develop consistent and concise conceptual models for domain-specific languages. Prior to developing tool support and model transformation, language specifications should at least consider to provide syntax, semantics, and notation for the different viewpoints [16].

Mulder [17] also acknowledged the need to validate the existing DEMO specification language (DEMOSL) prior to developing tool support. Using the meta-model definition presented by [18], metamodels should be sufficiently *complete* to describe all *set of models* (i.e. multiple viewpoints) that are allowed, rejecting models that are not valid. In addition, the metamodel should enable partial *transformation* of the model (e.g. from ontological to implementation level). With respect to the DEMO metamodels, Mulder [17] already suggested improvements regarding the multiple viewpoints evident in four aspect models. Since our first version of the DEMO-ADOxx tool only includes the construction model (CM), we elaborate within the next section on the updated metamodel for the CM.

2.1 DEMO Models and Metamodels

DEMO uses four linguistically based *aspect models* to represent the ontological model of the *organisation domain* of the enterprise, namely the *construction model* (CM), *process model* (PM), *action model* (AM), and *fact model* (FM) that exclude technology implementation details [19]. Each model is represented by different diagrams and tables, as illustrated in Fig. 1.

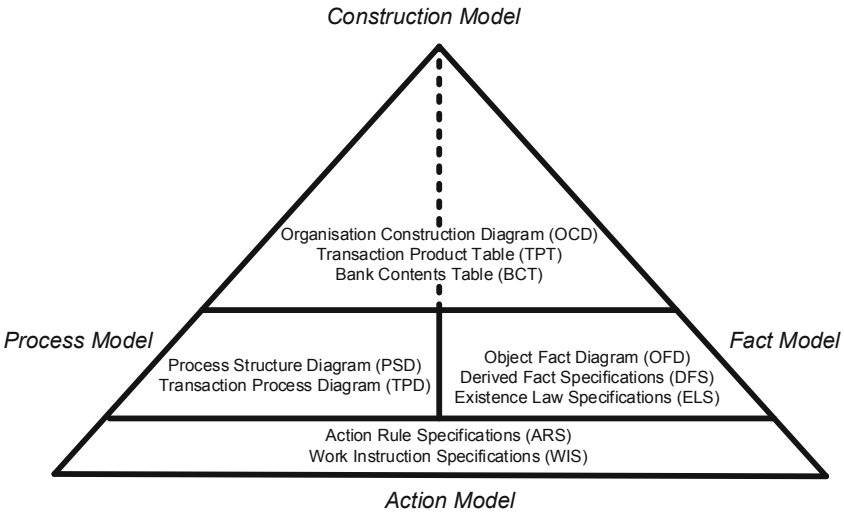

Fig. 1. DEMO aspect models with diagram types and tables, based on [19] and [20]

The ontological model is based on a key discovery that forms the basis of the aspect models, namely the identification of a *complete transaction pattern* that involves two *actor roles*, a *production act* (and fact), and *multiple coordination acts* (and facts) that

are performed in a particular order [19]. Although it is possible to identify three different *sorts* of a transaction kind (TK), i.e. *original*, *informational* and *documental*, the four aspect models primarily focus on the *original* sort. A TK can also be classified as an *elementary* TK when it is executed by only one actor role, or an *aggregate* TK (ATK) when it is executed by multiple actor roles. Also, an *actor role* can be classified as either an *elementary actor role* (EAR) when s/he executes one TK and a *composite actor role* (CAR) when s/he is the executor of more than one TK [19, 20].

The concepts that were discussed so far, as well as the relationships between concepts, are described via a metamodel presented in [19]. Mulder [17] identified several inconsistencies with regards to the CM, addressing the issues in [21]. Figure 2 presents an updated metamodel that incorporates the extensions suggested by Mulder [21]. Note that the *Scope of Interest* (SoI) is not modelled as a separate concept, since Mulder [21] argues that the SoI is equivalent to the CAR. The relationships and cardinalities in Fig. 2 signify modelling constraints when a modeller composes a CM. The constraints should also be incorporated in the modelling tool. As an example, a single relationship exists between *Transaction Kind* (TK) and *Aggregate Transaction Kind* (ATK) in Fig. 2. The relationship can be interpreted in a *forward direction* as: "One TK is contained in zero or many ATKs". The relationship interpretation of the *reverse direction* is: "One ATK contains one or many TKs".

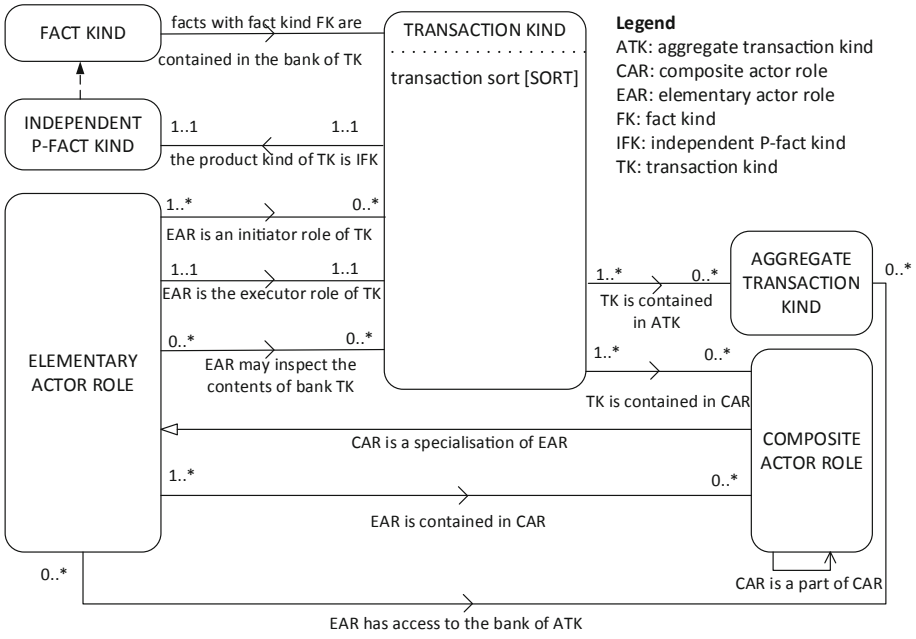

Fig. 2. DEMO construction model metamodel Version 3.7 [19] with extensions of [21]

2.2 The Demonstration Case

The demonstration case had to include the necessary complexity to ensure that a modeler would be able to construct a TPT (illustrated in Fig. 3) and an OCD (illustrated in Fig. 4) with all the relationships and cardinalities depicted in Fig. 2. Selecting a fictitious college as the universe of discourse, *some operations of the college* regarding the presentation of a new project-based module at the college, are incorporated, listed as *transaction kinds* in Fig. 3.

transaction ID	transaction kind	product ID	product kind
(T01)	supervisor allocation	(P01)	Supervisor-allocation is started
(T02)	project sponsoring	(P02)	the sponsorship of Project is done
(T03)	ip clearance	(P03)	the ip-clearance for Project is obtained
(T04)	module revision	(P04)	the revision of Module is done
(T05)	project control	(P05)	the project-control for Period is done
(T06)	internal project sponsoring	(P06)	the internal sponsorship of Project is done
(T07)	project involvement	(P07)	Project-involvement is started

Fig. 3. The TPT for a college, based on [19]

The reader is referred to [19] for a comprehensive introduction to the OCD and legend for concepts included in Fig. 2 and Fig. 4. In our demonstrating OCD, portrayed in Fig. 4, we assume that we only include TKs that are of the *original transaction sort,* in accordance with the guidelines presented by Dietz [20] to focus on the essential TKs. Based on the concepts declared in [19], we use **bold** style to indicate the type of construct and *italics* when referring to an instance of the construct (see Fig. 4).

Scope of Interest (SoI) indicates that the modeler analyses a particular scope of operations, namely *some operations at a college*. Given the SoI, Fig. 4 indicates that three **environmental actor roles** are defined, see the grey-shaded constructs *student*, *project sponsor* and *HR of project sponsor* that form part of the environment. Within the SoI, multiple **transaction kinds (TKs)** are linked to different types of **actor roles** via **initiation links** or **executor links**. As an example, *supervisor allocation* (*T01)* is a **TK** that is initiated (via an **initiation link**) by the **environmental actor role** *student (CA01)*. In accordance with [20], the *student (CA01)* is by default also regarded to be a **composite actor role** "of which one does not know (or want to know) the details". Since *T01* is linked to an **environmental actor role**, it is also called a **border transaction kind**. *T01* is executed (via the **executor link**) by the **elementary actor role** named *supervisor allocator (A01)*.

All the other actor roles in Fig. 4 within the **SoI** are **elementary actor roles**, since each of them is only responsible for executing one **transaction kind**. A special case of is where an **elementary actor role** is both the **initiator** and **executor** of a **transaction kind**, also called a **self-activating actor role**. Figure 4 exemplifies the **self-activating actor role** with *module reviser (A04)* and *project controller (A05)*. Since **actor roles** need to use facts created and stored in transaction banks, an **information link** is used

to indicate access to facts. As an example, Fig. 4 indicates that *project controller (A05)* has an **information link** to **transaction kind** *module revision (T04)*, indicating that the *project controller (A05)* uses facts in the transaction bank of *module revision (T04)*. It is also possible that **actor roles** within the **SoI** need to use facts that are created via **transaction kinds** that are outside the **SoI**. As an example, Fig. 4 indicates that **actor roles** within the **SoI** (called, *some operations at a college*) need to use facts that are created outside the SoI and stored in the transaction banks of **aggregate transaction kinds**, namely *person facts* of *AT01*, *college facts* of *AT02*, *accreditation facts* of *AT03*, *timetable facts* of *AT04* and *student enrollment facts* of *AT05*. According to Fig. 4, the *student enrollment facts* of **aggregate transaction kind** *AT05* are not accessed by any **actor roles**, which should be possible (according to the meta-model depicted in Fig. 2).

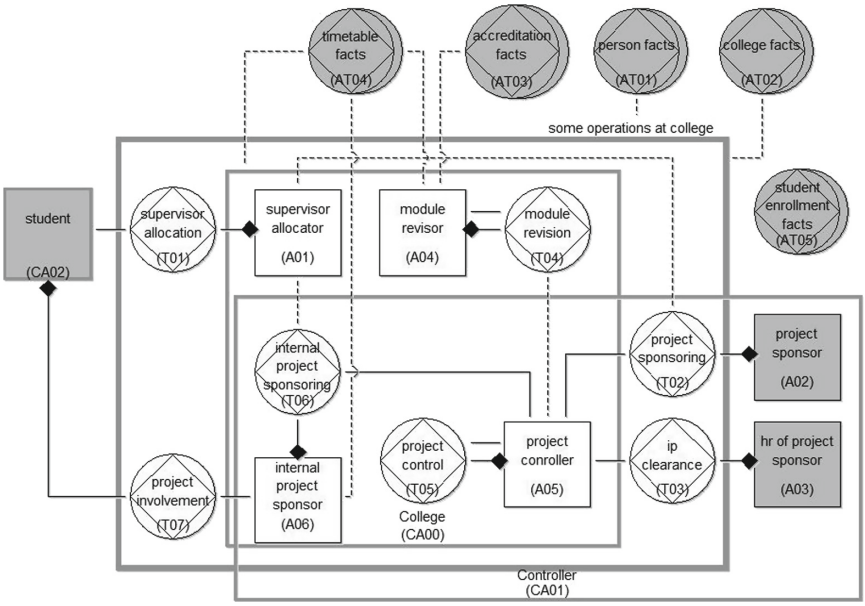

Fig. 4. The OCD for a college, based on [19]

Even though Fig. 4 only includes **elementary actor roles** within the **SoI**, it is possible to consolidate **elementary actor roles** within a **composite actor role**, where a composite actor role "is a network of transaction kinds and (elementary) actor roles" [20]. Figure 4 illustrates two **composite actor roles** within the **SoI,** namely *College (CA0)* and *Controller (CA01).* Both *CA00* and *CA01* encapsulate a number of transaction kinds and elementary actor roles.

3 Research Method

Applying *design science research* (DSR), we developed the DEMO-ADOxx modelling tool. According to Gregor & Hevner's [22] knowledge contribution framework, the modelling tool can be considered as an improvement, since the tool will be used for solving a known problem. Referring to the DSR steps of Peffers et al. [23], this article addresses the *five steps* of the DSR cycle in the following way:

Identify a Problem: In Sect. 4.1 we present minimal requirements for a useful DEMO modelling tool. Based on the requirements, we assess in Sect. 4.2 that existing DEMO modelling tools are inadequate.

Define Objectives of the Solution: In Sects. 4.3 and 4.4 we specify a new DEMO-ADOxx tool to address the requirements. We highlight that the DEMO-ADOxx tool only supports one of the four aspect models, namely the CM. Furthermore, the tool only incorporates two of the three CM representations, namely the OCD and TPT.

Design and Development: In accordance with the specification, we developed a DEMO-ADOxx tool, whose constructional components are presented in Sect. 5.

Demonstration: In accordance with the demonstration case, discussed in Sect. 2.2, we demonstrate the tool, highlighting its key feature, i.e. the transformation of a user-selected transaction kind of an OCD into a corresponding BPMN diagram.

Evaluation: Evaluation was restricted to internal testing, using the DEMO-ADOxx tool to model a more extensive case (than the demonstration case in Sect. 2.2). Individual test scenarios were created to validate each of the relationships and cardinalities illustrated in Fig. 2. The study excluded further evaluation, but Sect. 6 provides suggestions on further evaluating and extending the DEMO-ADOxx tool.

4 Requirements Elicitation and DEMO Tool Specification

A number of tools exist that support, to a limited extent, the creation of DEMO models. Before we compare these tools, the next section presents three categories of tool requirements from the perspective of a DEMO modeller.

4.1 DEMO Requirements

During the development of new software application systems, the analyst needs to consider three main categories of requirements, namely functional requirements, non-functional requirements, and design constraints [24]. In terms of the DEMO modelling tool, *functional requirements* regard the inputs, outputs, functions and features that are needed [24] (see R1 to R3 below). The *non-functional requirements* (see R4 and R5 below) incorporate the qualities of a system, such as performance, cost, security, and usability. The *design constraints* pose restrictions on the design of the system to meet technical, business, or contractual obligations. Next, we present initial requirements for

a DEMO tool, structured according to the first two categories. The purpose is to compare and evaluate the existing DEMO tools in terms of the following *minimum requirements* defined from the perspective of a lecturer teaching DEMO:

- R1: The DEMO tool should be *comprehensive* in supporting all of the DEMO aspect models, namely the CM, PM, AM and FM (refer to Fig. 1).
- R2: The DEMO tool should support the most recent published language specification, i.e. *DEMOSL 3.7* (see [19]) and the extensions that have been published (see [21]). The tool should be ready to accommodate future upgrades of the DEMO language.
- R3: The DEMO tool should facilitate *model transformations* to other modelling languages such as BPMN.
- R4: The DEMO tool should be available at *low cost*, especially for educational purposes.
- R5: The DEMO tool should be *usable*, i.e. user-friendly.

We used Nassar's *usability* requirements [25] to perform initial usability tests on some of the available DEMO tools:

- U1 Consistency: The system needs to be consistent in its actions, so that the modeller can get used to the system without constantly having to adapt to a new way of doing things. Consistency should apply to the way icons and commands are displayed and used.
- U2 User Control: The system should offer the user control in the way the model is built and run. This could include cancelling/pausing operations, undoing or redoing steps. The modeller should be able to foresee or undo errors.
- U3 Ease of learning: The system should be easy to learn for a new modeller. This is achieved by avoiding icons, layouts and terms that are unfamiliar to the modeller.
- U4 Flexibility: The system is expected to offer different ways to accomplish the same task so that the user experiences maximum freedom. Examples include shortcut keys, different icon options or even layout customisation.
- U5 Error Management: The system is expected to have built-in counter-measures to prevent mistakes by displaying error messages, warning icons or simply preventing incorrect placement of model elements.
- U6 Reduction of Excess: The system should avoid displaying unnecessary information or adding unnecessary functionality to the tool. The program should be functional and easy to understand.
- U7 Visibility of System Status: The user of the system should be aware of the status of the system at all times. For example, if a command does not occur instantaneously, then the system should inform the user of the delay.

4.2 Evaluating Existing DEMO Tools

In this section, we provide an overview of the existing tools, starting with a list presented in [26], adding Abacus and our ADOxx tool. In a first phase, we evaluated existing tools in terms of requirements R1 to R4 (see Table 1) using the following methods in order

of preference: (1) experimenting with the tools that were available; (2) contacting the tool owners for information about their tools; and (3) using the tool evaluation results of Mulder [26]. During a second phase, we tested the usability (R5) of four tools that were openly available (see Table 2).

Table 1. Evaluation results for functional and meta-model requirements

Requirement No ->	R1				R2	R3	R4	Legend	
Aspect model	CM	PM	AM	FM				●	Fully supports
Aris	●	○	○	○	○	○	◐		
CaseWise modeler	●	●	○	●	○	○	?	◐	Partially supports
Connexio knowledge system	●	○	●	●	○	○	○		
DemoWorld	●	●	○	●	○	○	◐	○	Does not support
EC-Mod	●	◐	○	○	○	○	○		
Plena	●	●	◐	◐	★	○	◐	★	Does not support, but vision for future
ModelWorld	●	●	○	●	○	○	●		
uSoft studio	●	○	○	○	○	○	?		
Visio	●	○	○	●	○	○	◐	?	Level of support is unclear
Xemod	●	●	○	●	○	○	○		
Abacus	●	◐	○	○	○	○	◐		
DEMO-ADOxx	●	○	○	○	●	●	●		

Table 2. Evaluation of usability requirements

	U1	U2	U3	U4	U5	U6	U7
Abacus	●	●	●	●	●	●	●
ModelWorld	●	○	●	○	●	●	○
Plena	●	●	◐	●	●	◐	●
DEMO-ADOxx	●	●	●	◐	●	●	●

Phase 1 Evaluation: In Table 1, we present evaluation results of existing DEMO tools with respect to R1 to R4, indicating the extent to which a specific tool meets a requirement, as explained in the legend of Table 1. R1, R2 and R4 were evaluated by Mulder [26] already. In his study he found that only Plena (of the studied tools) complies with R1 (i.e. support all four DEMO aspect models), none of the tools comply with R2 (i.e. supports the DEMOSL 3.7 specification language with extensions), and only ModelWorld complies with R4 (i.e. is available free of charge for academics and students).

Our ADOxx tool does not comply with R1, since the initial focus of the tool is to support the CM. For R2, the ADOxx tool supports DEMOSL 3.7 and the extensions. For R3 only the ADOxx tool supports transformations from DEMO models to other model types. Regarding R4, the ADOxx tool is free of cost for education purposes.

Phase 2 Evaluation: We had access to three of the existing DEMO modelling tools listed in Table 1, namely Abacus, ModelWorld and Plena. Using Nassar's *usability* requirements [25] listed in Sect. 4.1, we evaluated each of the three tools, also adding the DEMO-ADOxx tool, to gain some insights regarding their usability. The results are summarised in Table 2, indicating that three of the tools have usability *drawbacks*:

- *Modelworld* scored very low on U2 (User Control), U4 (Flexibility) and U7 (Visibility on System Status). Regarding U2 and U4, the modeller is unable to cancel any steps, undo any actions or navigate forwards and backwards. Basic keyboard shortcuts are not available to the user, such as the delete key. With reference to U7, ModelWorld offers no indication regarding the status of the system.
- *Plena* scored low on U3 (Ease of Learning), and U6 (Reduction of Excess). Plena is initially a challenge to use as it needs to be installed separately from Enterprise Architect and then imported as a plugin. Since Plena is a plugin to Enterprise Architect, some functionality is not applicable to DEMO.
- *DEMO-ADOxx* scored low on U4 (Flexibility), since the tool deviates from the standard drag-and-drop behaviour of other modelling tools. For this tool, a modeller needs to "left-click" on the construct in the template, dropping the construct by "left-clicking" within the modelling area on the right. It is possible to reason that the drag-and-drop behaviour is merely a behaviour-preference of one modeller and that *other modellers* will not highlight this as a usability deficiency.

The purpose of the evaluation was to provide an overview of the existing DEMO modelling tools to establish whether a new DEMO tool was needed. Even though existing tools are available, our main concern is that existing tools do not address requirements R2, R3 and R4. The new DEMO-ADOxx tool has been developed as a main deliverable for this study to address these three requirements. In terms of R1, the next section motivates the decision to initially set the scope to the DEMO CM.

4.3 DEMO Tool Specifications for the OCD and TPT

A qualitative analysis on DEMO aspect models, indicate that the CM, detailed by the PM, are useful for assigning responsibilities and duties to individuals [4]. The AM and FM "are necessary if you are going to develop or select applications" [4]. Since the conceptual knowledge embedded in the PM is similar to the BPMN collaboration diagram [12] and BPMN is widely adopted by industry [27, 28], the initial DEMO-ADOxx tool focuses on the CM. We exclude the PM, since the PM logic can be also represented by the industry-accepted notation BPMN. Our tool ensures consistent OCD-derived BPMN collaboration diagrams that incorporate the logic embedded in the *DEMO standard transaction pattern* as defined in [19].

We incorporated recent specifications regarding the OCD and TPT, as stated in [19] and [21], as well as BPMN 2.0 [29] for the first version of the DEMO-ADOxx tool. All of the existence rules, shown in Fig. 2 were implemented, except for one. The rule "facts with fact kind FK are contained in the bank of TK", indicated on Fig. 2, has not been incorporated in the DEMO-ADOxx tool, since it relates to the *bank contents table* (BCT), and the BCT relates to concepts that are used as part of the FM.

4.4 OCD-BPMN Transformations Specification

We identified four transformation scenarios that should be addressed by the DEMO tool. The specifications are excluded for the purpose of this article. Although the ADOxx-DEMO tool incorporates all four scenarios, we only include the *second scenario*, since this scenario already includes complexity of parent-and-part TKs. Referring back to the OCD depicted in Fig. 4, the four scenarios are as follows:

- *Scenario 1: Customer-initiated TK with no parts.* For this scenario, an **actor role** that is outside the **scope-of-interest**, initiates a **TK**. Also, the **TK** does not have any parts, i.e., the **executor** of the **TK**, is **not initiating** other **TKs**. Referring to Fig. 4, the TK labelled *T01* (*supervisor allocation*) is an example of this scenario. *T01* is initiated by the **actor role** *student*. The **executor** of *T01* is the *supervisor allocator.* Yet, the *supervisor allocator* does not initiate any other TKs as parts.
- *Scenario 2: TK is part of another TK.* For this scenario, the selected **TK** forms part of another **TK**. Referring to Fig. 4 the TK labelled *T07* (*project involvement*) is **initiated** by an **actor role** *A06* (*internal project sponsor*). Since the *internal project sponsor* is both the **executor** of *T06* (*internal project sponsoring*) and the **initiator** of *T07* (*project involvement*), *T07* is a part of *T06*.
- *Scenario 3: TK is self-initiating.* For this scenario, the selected **TK** is **initiated** and **executed** by the same **actor role**. Referring to Fig. 4, the TK labelled *T04* (*module revision*) is **initiated** and **executed** by *A04* (*module reviser*).
- *Scenario 4: TK has one or more parts.* For this scenario, the selected **TK** has one or more parts, i.e. the **actor role** that **executes** the **TK**, is also **initiating** one or more other **TKs**. Referring to Fig. 4 the TK labelled *T5* (*project control*) is executed by **actor role** *A05* (*project controller*). The same **actor role** *A05* (*project controller*) also **initiates** multiple other **TKs**, namely *T02* (*project sponsoring*), *T03* (*IP clearance*), and *T06* (*internal project sponsoring*).

5 Demonstration of the DEMO-ADOxx Tool in Use

The ADOxx platform, part the Open Models Laboratory (OMiLAB) digital ecosystem, is designed to support conceptualization and operationalization of conceptual modelling methods [30]. ADOxx allows a developer to create new modelling tools, or to extend existing ones to cater for any number of user requirements and customizations. The DEMO-ADOxx tool is realized as an OMiLAB project which enables free download[1].

[1] DEMO-ADOxx download: https://austria.omilab.org/psm/content/demo, last accessed: 09.04.2020.

5.1 Modelling and Validation Features

Figure 5 illustrates two main tool sections: (1) *Explorer* section - models (created before) are listed far left; and (2) *Modelling* section - OCD constructs are selected by "left-clicking" on the construct in the template, dropping the construct by "left-clicking" within the modelling area on the right. The relationships can be created either from dragging and dropping, or by using the *model assistant* which allows one to create a relationship directly from an existing construct in the model.

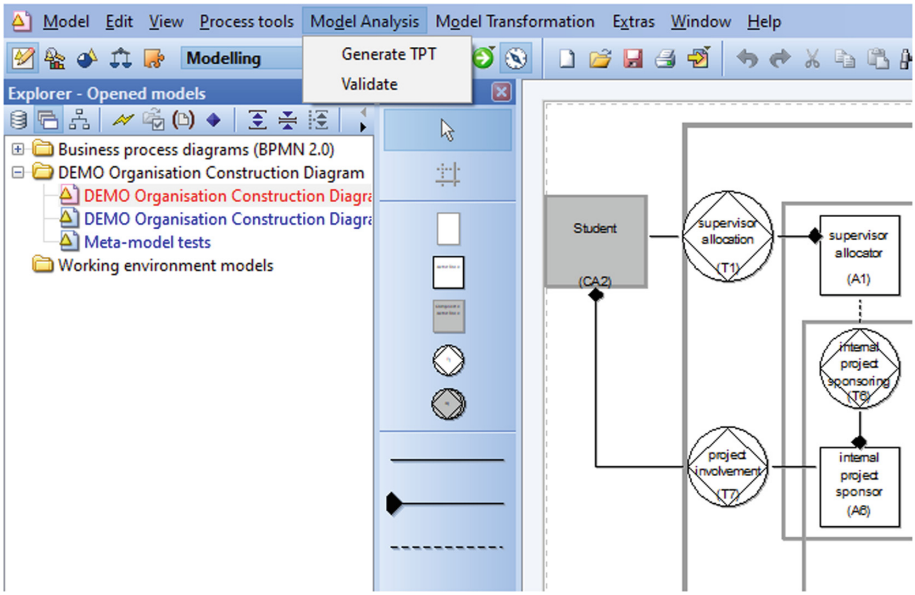

Fig. 5. The modelling interface for the DEMO-ADOxx tool

At the top of the screen are the menu options depicted. We implemented a *Model Analysis* menu that provides the option to either generate a TPT such as the one in Fig. 3, or to validate a model.

The *Validation* feature implemented each of the existence rules (relationships and cardinalities) presented in Fig. 2, except for one, as indicated before in Sect. 4.3. Figure 6 illustrates a validation table that communicates to the modeller: (1) The nature of a mistake in the model; and (2) The model constructs involved.

Based on the demonstration case discussed in Sect. 2.2, we used the new tool to generate an OCD (see Fig. 4) as well as a TPT (see Fig. 3) by utilizing the implemented semi-automatic model transformations of the DEMO-ADOxx tool.

5.2 Transformation Features

Selecting the *Model Transformation* menu option, the modeller can select one of the transaction kinds in the current OCD model. In our example the modeller selected *T07*

Description	Elements involved
Relationship with CAR needs to be more specific:	College
Actor role does not execute transaction kind:	supervisor allocator
No product kind defined	supervisor allocation
No product kind defined	project involvement

Fig. 6. Validation feature of the DEMO-ADOxx tool

(project involvement) that represents a *Scenario 2* transformation. The modeller also needs to specify the detail of interaction between parent-and-part TK's. In addition, cardinalities that exist between relevant parent-part structures, have to be specified by the modeller. As an example, Fig. 4 indicates that T07 is initiated by A06 (*internal project sponsor*). Yet, A06 is also the executor of T06 (*internal project sponsoring*). Therefore, T06 can only be requested when a T07 has made some progress through the sequence of coordination acts associated with the universal transaction pattern.

As indicated in Fig. 7, the modeller needs to indicate how T07 (project involvement) is initiated as a-part-of-T06 (internal project sponsoring), i.e. which one of the four basic coordination facts for T06 (requested, promised, stated or accepted) is a prerequisite for initiating T07. In addition, the modeller needs to indicate the cardinalities involved between one instance of the parent (T06) that generates a number of instances of the part (T07). For our demonstration (see Fig. 7), the modeller indicated that an instance of T06 has to be *stated* before T07 is *requested*. Also, one instance of T06 initiates zero-to-many (0..*) instances of T07. Since the transaction-progress of the parts may also regulate the transaction-progress of the parent, the modeller also has to indicate how the zero-to-many (0..*) part-instances (T06 instances) should all be *accepted* before the parent instance (T07 instance) can be *accepted*.

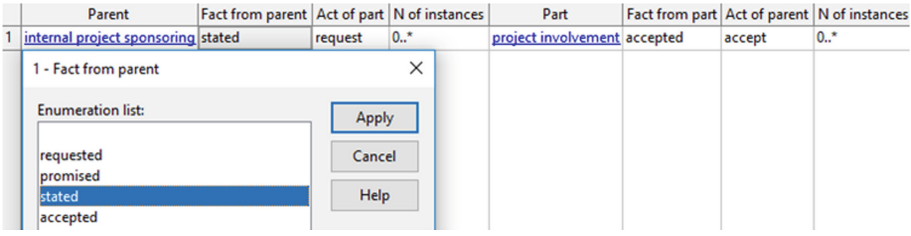

Fig. 7. User-interface to specify cardinalities for parent-part structures

Based on the modeller selections illustrated in Fig. 7, the DEMO-ADOxx tool automatically generates the corresponding BPMN collaboration diagram (see Fig. 8). The BPMN diagram (Fig. 8) presents the initiating actor role (*internal project sponsor*) as a BPMN pool and the executing actor role (*student*) as a BPMN pool. In accordance with transformation specifications (not detailed in this article), transaction pattern detail for the *standard pattern*, is depicted via BPMN concepts.

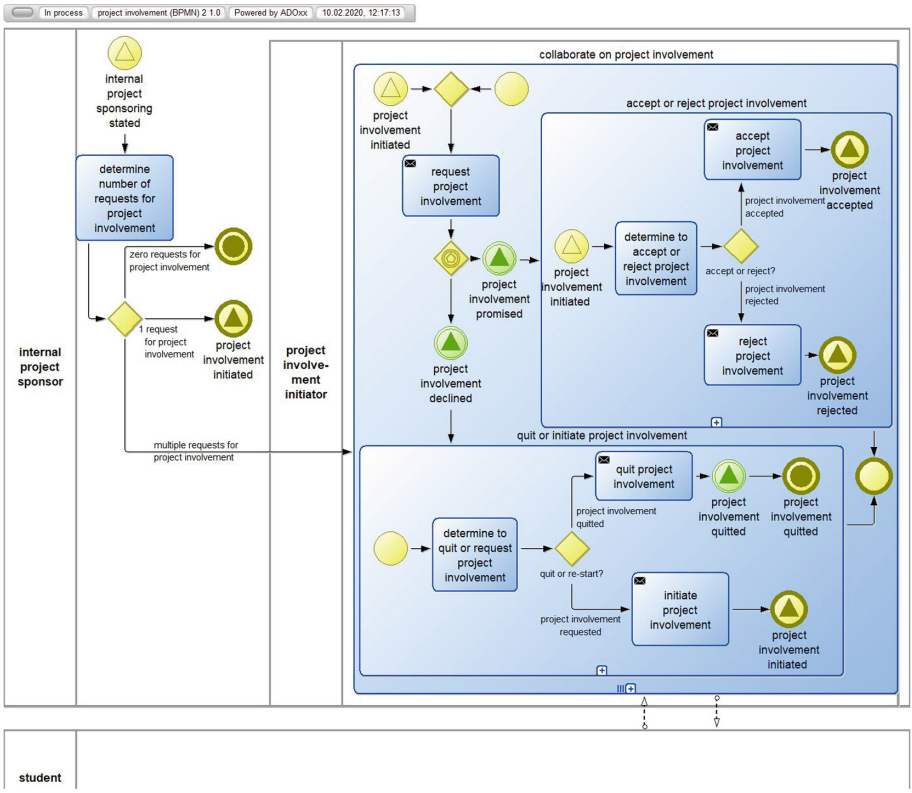

Fig. 8. BPMN collaboration diagram generated for T07 (project involvement)

6 Conclusions and Future Research

Our research indicated that existing DEMO modelling tools do not meet the minimum requirements. One of the key requirements is that the modelling tool needs to allow for model transformations, specifically transformations from a DEMO OCD to a BPMN collaboration diagram.

We have used two sets of specifications, (1) recent DEMO specifications from [19] and [21], and (2) OCD-BPMN transformation specifications, to develop a new DEMO-ADOxx tool to demonstrate the modelling and validation features, as well as the OCD-BPMN transformation feature.

The meta-model provided a good baseline for the DEMO-ADOxx tool. Yet, we accept that the meta-model will change in the future and these changes need to be accommodated by our tool in future. We still wait for feedback on the OCD-DEMO transformation specifications that will require further work on the DEMO-ADOxx tool. Realizing the tool as an open source project within the OMiLAB ensures that a community can take over future tool enhancements.

The demonstration case was useful in presenting the key features of the new DEMO-ADOxx tool. In terms of the *usability* requirements, additional evaluation is required. For future work, DEMO modellers will be involved during usability tests to inform further tool enhancements. In addition, a new version of DEMOSL will be released during 2020 and need to be incorporated within the DEMO-ADOxx tool.

References

1. Frank, U., Strecker, S., Fettke, P., Vom Brocke, J., Becker, J., Sinz, E.J.: The research field: modelling business information systems. Bus. Inf. Syst. Eng. **6**(1), 1–5 (2014). https://doi.org/10.1007/s12599-013-0301-5
2. Karagiannis, D., Mayr, H.C., Mylopoulos, J.: Domain-specific Conceptual Modeling: Concepts, Methods and Tools. Springer, Switzerland (2016). https://doi.org/10.1007/978-3-319-39417-6
3. Dietz, J.L.G.: Enterprise Ontology. Springer, Berlin (2006). https://doi.org/10.1007/3-540-33149-2
4. Décosse, C., Molnar, W.A., Proper, H.A.: What does DEMO do? A qualitative analysis about DEMO in practice: founders, modellers and beneficiaries. In: Aveiro, D., Tribolet, J., Gouveia, D. (eds.) EEWC 2014. LNBIP, vol. 174, pp. 16–30. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-06505-2_2
5. Recker, J., Indulska, M., Rosemann, M., Green, P.: How good is BPMN really? Insights from theory and practice. In: Ljungberg, J., Andersson, M. (eds.) Proceedings 14th European Conference on Information Systems, ECIS, pp. 1582–1593 (2006)
6. Van Nuffel, D., Mulder, H., Van Kervel, S.: Enhancing the formal foundations of BPMN by enterprise ontology. In: Albani, A., Barjis, J., Dietz, J.L.G. (eds.) CIAO!/EOMAS -2009. LNBIP, vol. 34, pp. 115–129. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01915-9_9
7. de Kinderen, S., Gaaloul, K., Proper, H.A.: On transforming DEMO models to ArchiMate. In: Bider, I., et al. (eds.) BPMDS/EMMSAD -2012. LNBIP, vol. 113, pp. 270–284. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-31072-0_19
8. Pijpers, V., Gordijn, G., Akkermans, H.: E3alignment: exploring inter-organizational alignment in networked value constellations. Int. J. Comput. Sci. Appl. **6**(5), 59–88 (2009)
9. Ettema, R., Dietz, J.L.G.: ArchiMate and DEMO – mates to date? In: Albani, A., Barjis, J., Dietz, J.L.G. (eds.) CIAO!/EOMAS -2009. LNBIP, vol. 34, pp. 172–186. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01915-9_13
10. Caetano, A., Assis, A., Tribolet, J.: Using DEMO to analyse the consistency of business process models. In: Moller, C., Chaudhry, S. (eds.) Advances in Enterprise Information Systems II, pp. 133–146. Taylor & Francis Group, London (2012)
11. Heller, S.: Usage of DEMO methods for BPMN models creation. Czech Technical University in Prague (2016)
12. 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.) EEWC 2017. LNBIP, vol. 284, pp. 85–98. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-57955-9_7
13. France, R., Rumpe, B.: Model-based development. Softw. Syst. Model. **7**(1), 1–2 (2008)
14. Cicchetti, A., Ciccozzi, F., Pierantonio, A.: Multi-view approaches for software and system modelling: a systematic literature review. Softw. Syst. Model. **18**(6), 3207–3233 (2019). https://doi.org/10.1007/s10270-018-00713-w
15. Bork, D.: A development method for conceptual design of multi-view modeling tools with an emphasis on consistency requirements. University of Bamberg (2016)
16. Grundy, J., Hosking, J., Li, K.N., Ali, N.M., Huh, J., Li, R.L.: Generating domain-specific visual language tools from abstract visual specifications. IEEE Trans. Softw. Eng. **39**(4), 487–515 (2013)
17. Mulder, M.A.T.: Validating the DEMO specification language. In: Aveiro, D., Guizzardi, G., Guerreiro, S., Guédria, W. (eds.) EEWC 2018. LNBIP, vol. 334, pp. 131–143. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-06097-8_8
18. Aßmann, U., Zschaler, S., Wagner, G.: Ontologies, meta-models, and the model driven paradigm. In: Calero, C., Ruiz, F., Piattini, M. (eds.) Ontologies for Software Engineering and Software Technology, pp. 249–273. Springer, Heidelberg (2006). https://doi.org/10.1007/3-540-34518-3_9
19. Dietz, J.L.G., Mulder, M.A.T.: DEMOSL-3: demo specification language version 3.7. SAPIO (2017)
20. Perinforma, A.P.C.: The Essence of Organisation, 3rd ed. Sapio (2017). www.sapio.nl
21. Mulder, M.A.T.: Towards a complete metamodel for DEMO CM. In: Debruyne, C., Panetto, H., Guédria, W., Bollen, P., Ciuciu, I., Meersman, R. (eds.) OTM 2018. LNCS, vol. 11231, pp. 97–106. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-11683-5_10
22. Gregor, S., Hevner, A.: Positioning and presenting design science research for maximum impact. MIS Q. **37**(2), 337–355 (2013)
23. Peffers, K., Tuunanen, T., Rothenberger, M., Chatterjee, S.: A design science research methodology for information systems research. J. MIS **24**(3), 45–77 (2008)
24. Leffingwell, D.: Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise. Addison-Wesley, New Jersey (2011)
25. Nassar, V.: Common criteria for usability review. Work **41**(Suppl 1), 1053–1057 (2012)
26. Mulder, M.A.T.: Enabling the automatic verification and exchange of DEMO models. Ph.D. thesis (n.d.)
27. Grigorova, K., Mironov, K.: Comparison of business process modeling standards. Int. J. Eng. Sci. Manag. Res. **1**(3), 1–8 (2014)
28. Recker, J., Wohed, P., Rosemann, M.: Representation theory versus workflow patterns – the case of BPMN. In: Embley, David W., Olivé, A., Ram, S. (eds.) ER 2006. LNCS, vol. 4215, pp. 68–83. Springer, Heidelberg (2006). https://doi.org/10.1007/11901181_7
29. Object Management Group: Business process model & notation. https://www.omg.org/bpmn/. Accessed 30 May 2019
30. Bork, D., Buchmann, R.A., Karagiannis, D., Lee, M., Miron, E.-T.: An open platform for modeling method conceptualisation: the OMiLAB digital ecosystem. Commun. AIS **44**(32), 673–697 (2019)