Identifying scenarios to guide transformations from DEMO to BPMN

Marné De Vries and Dominik Bork

Appeared in:

D. Aveiro, D. Guizzardi, R. Pergl, H. Proper (Eds.): 10th Enterprise Engineering Working Conference (EEWC 2020), pp. 92 – 110

https://doi.org/10.1007/978-3-030-74196-9_6

www.model-engineering.info

Identifying scenarios to guide transformations from DEMO to BPMN

Marné De Vries[1][0000-0002-1715-0430], Dominik Bork[2][0000-0001-8259-2297]

[1]University of Pretoria, Department of Industrial and Systems Engineering, South Africa
[2] TU Wien, Business Informatics Group, Vienna, Austria
Marne.DeVries@up.ac.za; dominik.bork@tuwien.ac.at

Abstract The heterogeneity in enterprise design stakeholders and models generally demands for consistent and efficient transformations of enterprise design knowledge between different conceptual modelling languages. A systematic process and precise model transformation specifications are a prerequisite for realizing such transformations. The Design and Engineering Methodology for Organizations (DEMO) approach represents the organization design of an enterprise in four linguistically based, semantically sound aspect models. The Business Process Model and Notation (BPMN) on the other hand enables more flexibility in creating models and benefits from wide adoption in industry, the execution of processes e.g., by simulations, and the availability of proper tooling. A transformation of DEMO models into BPMN models is thus desirable to avail of both, the semantic sound foundation of DEMO and the wide adoption and execution possibilities of BPMN. Previous research already developed some principles and practices for transforming DEMO models into BPMN models, based on DEMOSL 3.7. This study focuses on the latest DEMO language specification, DEMOSL 4.5, since we believe that more clarity is required to specify consistent, well-motivated transformation specifications. We present a list of main requirements for developing transformation specifications to transform concepts represented in a Coordination Structure Diagram and Process Structure Diagram of DEMO into corresponding concepts in a BPMN collaboration diagram. The article makes three contributions: (1) Generic requirements for developing DEMO-to-BPMN transformation specifications; (2) Nine transformation scenarios that are validated by multiple demonstration cases; and (3) A comprehensive college case that demonstrates all transformation scenarios.

Keywords: DEMO, BPMN, model transformation, organization design.

1. Introduction

We live in an era where enterprises increasingly depend on digital technologies to enhance the speed of product development/service provision as well as communication with customers and collaboration within ecosystems. Within this context, agile (re-)engineering of an enterprise, involving heterogeneous stakeholders, there is a need to represent an enterprise using different modelling languages. The problem is that these representations (i.e., conceptual models) are often based on different meta-models that

need to be kept consistent [1]. Horizontal consistency refers to consistency between models at the same development phase, such as the analysis phase, whereas vertical consistency refers to consistency between models across different phases [2]. A recent systematic literature review (SLR) highlighted that the majority of studies focus on vertical integration and raised the need to also develop approaches that address horizontal consistency amongst models [3]. Another SLR [4] on existing tools that support horizontal transformations, indicate that only 7 out of 40 prominent modelling approaches support multi-view artefact creation, whereas only 6 of these 7, provide semi-automated support with for instance "wizards in the modelling environment".

Enterprises are organized complexities, i.e., too organized for applying statistics to understand their behavior and too complex to being studied by analytical methods [5]. Different stakeholders use different cognitive perspectives to understand and represent the enterprise [6]. Some stakeholders prefer structural thinking (demonstrated in DEMO's Coordination Structure Diagram (CSD) [5]), whereas others prefer flow thinking (demonstrated in BPMN [7]). Table 1 compares DEMO models to BPMN models in terms of their focus, strengths and weaknesses.

Table 1. Comparing focus, strenghs and weaknesses of DEMO versus BPMN

DEMO focus	**BPMN focus**
Often used *top-down* to represent the ideal design of a new enterprise [8].	Often used for *bottom-up* analysis of implemented processes as a starting point for re-design [7].
Represent *no implementation* [5].	Represent *implementation* [7].
Used to *reduce complexity*, extracting the essence of enterprise operation [5].	Used to *elucidate complexity*, since the models represent implementation.
DEMO strengths	**BPMN weaknesses**
Comprehensive representation of human collaboration during enterprise operation, since collaboration is based on the PSI theory, i.e., acknowledging a *complete* transaction pattern that exists between two actor roles [5].	The style and practice associated with BPMN modelling do not acknowledge the existence of the PSI theory [9] and hence models may be *incomplete* in representing the complete transaction pattern.
Provides a means for *clear scoping*, since CSDs use composite transactors to indicate where further elaboration of the model is needed [5].	Scope of a BPMN diagram is *unclear*, unless specified in a narrative that is associated with the diagram [7].
DEMO weaknesses	**BPMN strengths**
Although management appreciates the compact representation of enterprise operations, shown in the CSD, experienced modelers are needed. *Additional methods* are required to facilitate collaborative developments with relevant stakeholders [10].	Due to their descriptive and expressive abilities, BPMN models are *widely adopted* [11].

Based on their different foci, strengths and weaknesses we believe that enterprises will benefit by using both modelling languages, but also ensure horizontal consistency between the DEMO and BPMN models. DEMO models focus on intellectual manageability and reduction of complexity, representing business processes in a compact manner by focusing on a company's operations [5]. BPMN, on the other hand, allows for detailed descriptions of business processes, adding implementation logic that facilitates process execution [7]. The ontological model of the enterprise, as provided

by DEMO, is needed, since it exhibits four qualities: *comprehensiveness*, *coherency*, *consistency* and *conciseness* [5]. Yet, conceptual models that encounter for technological issues are also needed, since they allow for processing, such as simulation and workflow execution, as demonstrated by BPMN-based industrial tools. Due to their descriptive and expressive abilities, BPMN models are widely adopted [11]. Although BPMN models allow for flexibility to express knowledge about a process, modelers who are not properly guided are likely to produce ambiguous, inconsistent and incomplete models [12].

Researchers have already identified the need for transforming DEMO models into other models for various reasons, e.g.,: (1) transforming concepts from DEMO to concepts contained within the ArchiMate business layer meta-model for the purpose of modelling the essential aspects of an enterprise first in DEMO, followed by a transformation into technological realization and implementation models [12]; (2) transforming the DEMO process models into Petri net models to facilitate simulation [12; 13]; (3) transforming DEMO action models into BPMN models [14; 15]; and (4) transforming DEMO organization construction diagrams into BPMN collaboration diagrams to semi-automate DEMO-to-BPMN transformations [16]. As indicated in Fig. 1, the ADOxx-based tool, called DMT [17] (downloadable from: http://austria.omilab.org/psm/content/demo) already facilitates transformation from an existing organization structure diagram (OCD) to a BPMN collaboration diagram. When a modeler selects a transaction kind (in this example T01) for transformation, the transformation script identifies the *student* as the initiating actor role and the *supervisor allocator* as the executing actor role. The transformed BPMN diagram, thus includes a *student pool* and a *supervisor allocator pool*.

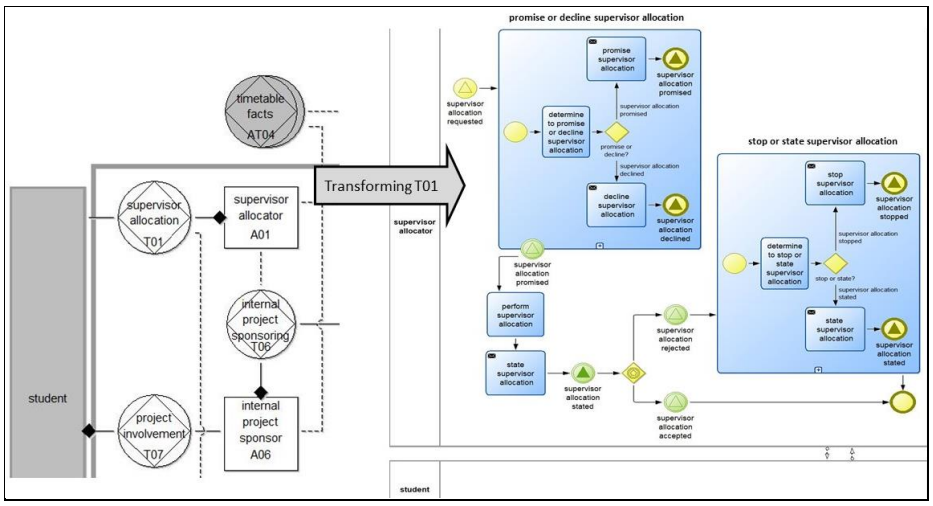

Fig. 1 DEMO-to-BPMN transformation, adapted from [16]

The transformation specifications that were used to generate the BPMN diagram shown in Fig. 1, were based on the *standard transaction pattern*, ensuring that the model explicitly incorporates coordination acts/facts that form part of the standard pattern, i.e. including: request, promise, decline, state, accept, and reject acts/facts. Without proper

guidance in terms of the standard pattern, BPMN models tend to be incomplete [18]. As an example, an unguided modeler may omit some of the acts (that form part of the standard pattern) when designing enterprise operations, e.g. omitting the *promise* and *decline* acts. An essential part of coordination will not be supported when the model is further refined for implementation. With no explicit *promise* or *decline* built into the design of the process, an instance of this process implies that a student, requesting supervisor allocation, receives no feedback in terms of the status. Was the request valid? Was the request declined?

This paper extends the DEMO model transformation research stream with the objective of using a DEMO Coordination Structure Diagram (CSD), validated with a Transactor Product Table (TPT), and the transaction-interaction logic between transaction kinds, represented on the Process Structure Diagram (PSD) to derive consistent BPMN collaboration diagrams. Using Design Science Research (DSR), we indicate in Section 3, that a set of requirements need to be identified prior to the development of valid transformation specifications. Previous work already demonstrated the possibility of using a tool to transform some DEMOSL 3.7 concepts to BPMN 2.0 concepts [19]. The previous transformations were based on incomplete transformation specifications, including only four transformation scenarios. The objective of this article is thus to elicit *main requirements for a comprehensive model transformation specification*, and, consequently, to define a *comprehensive set of nine DEMO to BPMN transformation scenarios*. We present these transformation scenarios and validate them in multiple demonstration cases.

The article is structured as follows. In Section 2, we provide background on the DEMO aspect models, motivating the need to develop a comprehensive set of TK-BPMN transformation scenarios. Section 3 presents Design Science Research (DSR) as an appropriate research method, suggesting that a comprehensive demonstration case is developed (presented in Section 4) as well as *main requirements* for transformation (presented in Section 5). We present our main contribution, *nine transformation scenarios* and their validation in Section 6, concluding with suggestions for future research in Section 7.

2. Background Theory

Modelling an enterprise, is not a trivial task. The emerging discipline of enterprise engineering (EE), acknowledges the existence of several enterprise design domains [20]. De Vries [21] suggests that four main enterprise design domains exist: (1) Organization; (2) ICT; (3) Infrastructure (including facilities); and (4) Human skills & know-how. The organization design domain is a social system that includes human beings as system elements. The human beings form relationships due to their interactions and communications when they perform production acts [5]. In this article we focus primarily on the organization design domain and its representation using DEMO aspect models and BPMN models.

In representing the organization design domain, the ontological model of an enterprise is based on the *performance in social interaction* (PSI) theory that provides a universal building block of enterprise organization [5]. The PSI theory identified transaction

patterns, each involving two actor roles, a production act (and fact), and multiple coordination acts (and facts) that are performed in a particular sequence. In terms of the identified pattern, Dietz and Mulder [5] indicated that a *complete* transaction pattern exists to represent the possible coordination acts (and facts) that describe interactions between two actor roles for a particular transaction. Most of the transactions follow the *basic* transaction pattern (i.e., the happy flow), where two actor roles (i.e., *initiator* and *executor*) are in consent to each other's intentions when following four coordination acts in sequence, namely *request*, *promise*, *declare*, and *accept*. Yet, when the actor roles do not comply with each other's intentions, they follow a *standard* transaction pattern, which allows for a *decline act* (instead of a promise act) and a *reject act* (instead of an accept act). It is possible that actor roles need to revoke some of the coordination acts that were already performed. Once a *request act* was performed, the initiator may have second thoughts, requesting to revoke the initial *request act*. Likewise, the *promise* act, *declare* act and *accept* act may be revoked. The *complete* transaction pattern extends the *standard* pattern with four revocation patterns [5].

Even though every transaction follows a path through a *complete* transaction pattern, transactions differ in the kind of product they produce. A transaction is thus an instance of a transaction kind (TK) executed by an actor role (AR), to produce a product kind (PK). The three concepts represent two different facets of the organization domain. Whereas TKs and ARs represent the coordination world, the PKs represent the production world.

Since our main objective is to transform DEMO concepts into BPMN concepts we need to relate the conceptual schema and its corresponding symbolic formalism of DEMO to that of BPMN. The conceptual schema of DEMO indicates that any enterprise can be represented by four aspect models that include a Process Model (PM), Action Model (AM), Cooperation Model (CM), and Fact Model (FM). Each model is represented by different diagram types and tables.

Since a BPMN collaboration diagram (CD) focuses on coordinating activities performed by actors or departments [7], it should be possible to relate the concepts included in the CD to concepts and logic included in aspect models that focus on coordination. As illustrated in Fig. 2 (left-hand side), the CM, PM and AM focus on coordination. According to [5], the AM is the most detailed model of the four aspect models. If the main objective is to perform a comprehensive transformation of coordination logic from DEMO to BPMN, we need to use the AM to generate a BPMN CD. Yet, from a management perspective the CM is more useful, since it provides bird's-eye view of coordination structures [5; 22]. Thus, from a pragmatic viewpoint, we want to extract coordination logic from appropriate aspect models that are already used in industry. We envision a tool that would enable semi-automatic transformation from transaction kinds that feature on a validated CM to a BPMN CD. Starting with the well-validated CM, and a means to indicate how events in one TK restrict transaction progress for other TKs (usually depicted on the PM), a modeler should select a single TK that needs to be transformed into a corresponding BPMN CD. The diagrams/tables that represent the CM and PM are therefore the most relevant for the envisioned transformations. In this article we demonstrate how we intend to transform knowledge from the CM, represented by the Coordination Structure Diagram (CSD) and the Transactor Product Table (TPT), as well as the PM, represented by the Process Structure Diagram (PSD).

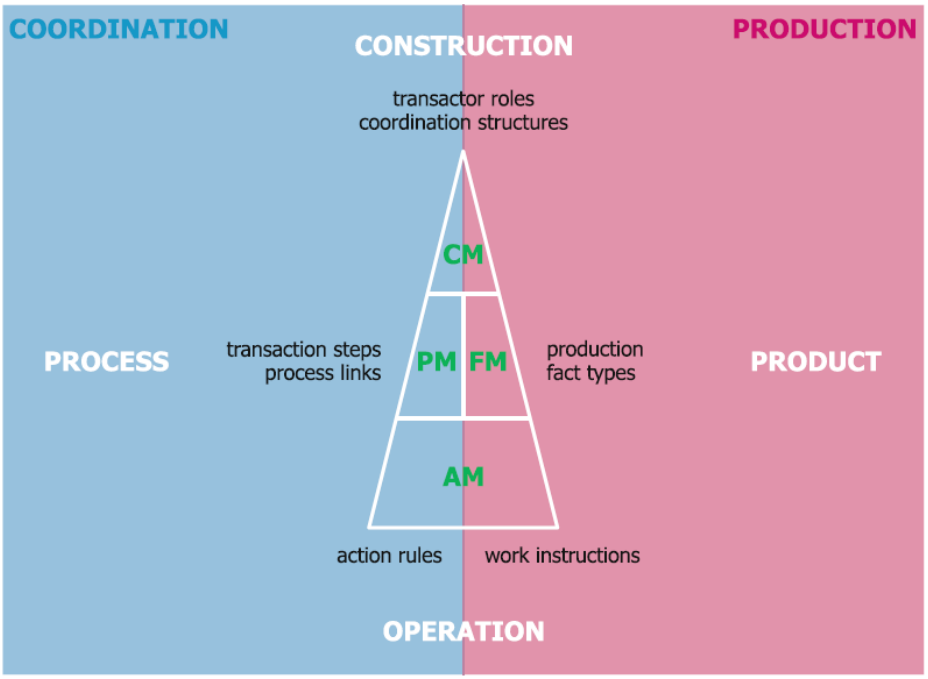

Fig. 2. Aspect models from [5]

We exclude the action model (AM) from our transformation scope, conceding that the transformed BPMN CD's will exclude detailed action rules that guide the actor roles. Again, from a pragmatic viewpoint, the detail of the AM may simply not be available. Yet, in Section 7 we also suggest that our transformation specifications need to allow for extensions points to incorporate detailed action rules. We also exclude the fact model (FM) from our transformation, since it represents the state space and the transition space of an organization's *production world* [5]. Instead, this article focuses primarily on the transformation space of the *coordination world*.

Unlike most common modelling languages, DEMO comes with a built-in quality of completeness with respect to the applied *patterns*. DEMO thereby guides the modeler toward the creation of complete models that cover all coordination acts and facts. Even though DEMO comes with a Process Model (discussed in section 4.1), the DEMO Process Model is not as widely used as the Cooperation Model [22]. In our approach, we aim at amplifying the respective and complementary strengths of both modelling languages, DEMO and BPMN, i.e., the completeness and sound foundation of DEMO models with the wide-adoption and execution possibilities of BPMN process models.

3. Research Method

This study applies the *five phases* of the Design Science Research Methodology [23] as follows:

Problem: Although BPMN is a well-adopted modelling standard applied by industry to model business processes, additional guidance is required to create consistent and executable BPMN models [24]. Mraz *et al.* [25] and Rodrigues [15] have already developed some principles and practices for transforming OCD constructs (based on DEMOSL 3.0) and the underlying transaction patterns for *transaction kinds* into BPMN models. Others [19] already demonstrated the possibility of using a tool to transform some DEMOSL 3.7 concepts to BPMN 2.0 concepts, but a sub-set of invalid transformations are generated when using the tool, since the transformation specifications were based on a set of incomplete transformation scenarios.
Solution Objectives: We believe that more clarity is required to specify consistent, well-motivated transformation specifications, based on DEMOSL 4.5. First, we need a demonstration case that is comprehensive enough in terms of the concepts that form part of the CSD and PSD, as defined in DEMOSL 4.5. Then, a set of main requirements should be elicited, prior to the development of detailed TK-BPMN transformation specifications. In addition, a comprehensive set of transformation scenarios should be developed, based on a comprehensive case.
Development: The solution objectives are addressed by presenting a comprehensive demonstration case (in Section 4), a list of main TK-BPMN transformation requirements (in Section 5) and nine transformation scenarios (in Section 6.1).
Demonstration: We validate the comprehensiveness of the nine transformation scenarios by applying the scenarios to multiple cases (in Section 6.2).

4. College Demonstration Case

In order to explain the DEMO language, we start with the instance level, i.e., a model of a fictitious enterprise where the scope-of-interest is *some operations at a college*. We introduce two of DEMO's four aspect models (PM and CM) and the symbolic formalism that is used to express the essence of *some operations at a college*. Our objective is to highlight concepts that will be used to distinguish between nine transformation scenarios that are presented later in Section 6.1.

4.1. The Process Model

The Process Model is the ontological model of the state space and the transition space of its coordination world and is depicted by two diagrams: (1) the Transaction Pattern Diagram (TPD), based on the *complete* transaction pattern, and (2) the Process Structure Diagram (PSD) [5]. The same pattern may apply to different *kinds of transactions*. Each *transaction* is thus an instance of a particular *transaction kind (TK)*, and the transaction produces a *product* that is an instance of a *product kind (PK)*. As an example from Fig. 3, the TK 06 (named *internal project sponsoring*) may produce PK06 of which the name is indicated in Table 2 (i.e. *the internal sponsorship of [project] is done*). The PK incorporates variables (indicated in square brackets) as placeholders for entities. Thus, for PK06, [project] is a placeholder that is used to differentiate between projects that are created from different instances of TK06.

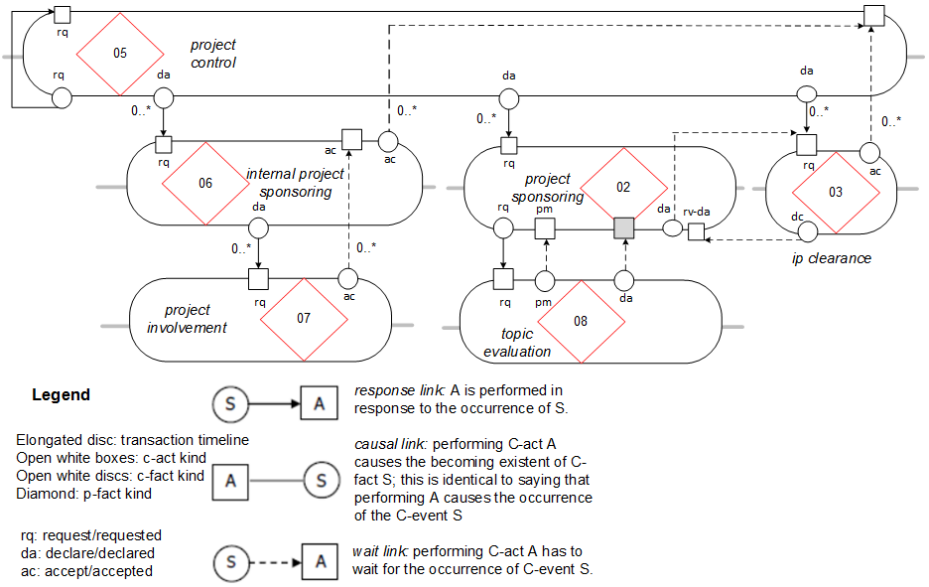

Fig. 3. The PSD indicates parent-part structures

When we consider an enterprise, e.g., our *fictitious college* that offers a project-based course to students, the enterprise may be responsible for various different TKs associated with this project-based course offering. For each of the TKs, it is possible to identify the *initiator* and *executor* that are coordinating their actions to realize a particular PK for the particular TK. Yet, the TKs are not detached from one another. The *process structure diagram* (PSD) is useful to delineate transitions in the coordination world, e.g., indicating how TK06 (e.g., *internal project sponsoring* is *declared*) has a response (indicated with a *response link*) on the transition of TK07 (i.e., zero-to-many instances of *project involvement* is *requested*). Also, the *accept* act of T06 (i.e., of *internal project sponsoring*) has to wait (indicated with a *wait link*) for the occurrence of zero-to-many *accept facts* from T07 (i.e. from *project involvement*).

Fig. 3 indicates that a hierarchy exists between TKs, implying that TK07 (i.e. *project involvement*) is a part of the parent TK06 (i.e. *internal project sponsoring*). The PSD *only* includes those coordination acts/facts from the interacting TKs that control progress of the two TKs. Thus, the *promise* act/fact of TK06 (in Fig. 3) is *not shown*, since the *promise* act/fact does not have a transitional effect on TK07.

4.2. Problems Identified in Previous TK-based Transformations to BPMN

Having discussed the purpose of a PSD, we highlight four problems with previous TK-BPMN transformations and the scenarios that were used in [19]. We have also adapted our demonstration (already reflected in Fig. 3) to explicate these problems.

The *first problem* reflects that the transformation specifications failed to represent the relevant parts of the transaction pattern when the initiating fact from the parent TK

changes. With reference to Fig. 3, when a modeler selected TK07 for transformation, the BPMN CD recognized that TK07 is a part of TK06 and that the TK07-process should start with the fact TK06/declared. In accordance with Fig. 3, the transformed BPMN diagram did not have to include the TK06/execute act, since TK06 is executed before TK06 is declared. Yet, if the PSD changed, indicating that TK06/requested is a prerequisite for TK07/request and TK06/requested precedes the TK06/executed diamond, then TK06/execute should be reflected in the transformed BPMN diagram.

The *second problem* reflects that the specifications did not address multiple *response links* and *causal links* between parent-part structures that are usually represented by the PSD. Therefore, when we applied the TK-BPMN transformations to the Rent-A-Car case (of [26]), where three interactions exist between *rental concluding* and *rental payment*, the transformation failed. DEMOSL 4.5 [27] still allows for more than two interactions. Hence, for our demonstration case PSD, we have included multiple interactions between TK02 and TK08, as indicated in Fig. 3.

The *third problem* reflects that for parent-part interaction, the initial demonstration case in [19] only incorporated coordination facts from the *basic* pattern and not the *standard* pattern. Therefore, when we applied the TK-BPMN transformations to the Rent-A-Car case (of [26]), where a *reject* fact of *car returning* (from the *standard* pattern) initiates *penalty payment*, the transformation could not be executed, since the modeler could not select a *reject* fact from the list of interaction acts/facts. Our PSD (see Fig. 3) now includes a *decline* fact (from the *standard* pattern) for TK03.

The *fourth problem* reflects that for parent-part interaction, the initial demonstration case in [19] did not include interaction (interimpediment structures) between TKs, since interimpediment structures were only included in a more recent DEMO specification [5].

4.3. The Cooperation Model

The *cooperation model* (CM) provides a concise representation of enterprise operations and consists of three representations: (1) a Coordination Structure Diagram (CSD), (2) a Transactor Product Table (TPT), and (3) a Bank Contents Table (BCT). The TPT indicates that every transaction kind (TK) produces a product kind (PK) via an executing actor role (AR). The BCT indicates that every TK produces/uses several independent/dependent facts during the execution of the TK. The CSD also represents TKs, but in a different format. Since every elementary TK can only be executed by one AR, the TK and AR are consolidated into a transactor role (TAR). As an example, a TAR named *supervisor allocator* implies that AR named *supervisor allocator* is the executor of the TK named *supervisor allocation*.

We explain the constructs of the TPT and CSD using a fictitious college as a demonstration case, presented in Table 2 and Fig. 4 respectively.

Table 2. TPT for the SoI defined for the college

transaction kind	product kind	executor role
TK01 supervisor allocation	PK01 [supervisor allocation] **is** done	AR01 supervisor allocator
TK02 project sponsoring	PK02 **the** sponsorship **of** [project] **is** done	AR02 project sponsor
TK03 ip clearance	PK03 **the** ip-clearance **for** [project] **is** done	AR03 ip clearer
TK04 module revision	PK04 module revision **for** [year] **is** done	AR04 module revisor
TK05 project control	PK05 project control **for** [year] **is** done	AR05 project controller
TK06 internal project sponsoring	PK06 **the** internal sponsorship **of** [project] **is** done	AR06 internal project sponsor
TK07 project involvement	PK07 [project involvement] **is** done	AR07 project involver
TK08 topic evaluation	PK08 the topic evaluation of [project] **is** done	AR08 topic evaluator
TK09 course registration	PK09 [course registration] **is** done	AR09 course registrar
TK10 course payment	PK10 [course registration] **is** paid	AR10 course payer
TK11 course admission	PK11 [course admission] **is** done	AR11 course admitter
TK12 bursary allocation	PK12 [bursary allocation] **is** done	AR12 bursary allocator
TK13 study-pack sale completion	PK13 [study-pack sale] **is** completed	AR13 study-pack sale completer
TK14 study-pack sale preparation	PK14 [study-pack sale] **is** prepared	AR13 study-pack sale preparer
TK15 study-pack sale payment	PK15 [study-pack sale] **is** paid	AR15 study-pack sale payer
TK16 study-pack sale selection	PK16 [study-pack sale] **is** selected	AR16 study-pack sale selector
TK17 study item buying	PK17 [study item] **is** bought	AR17 study item buyer
TK18 course design	PK18 [course] **is** designed	AR18 course designer
TK19 candidate evaluation	PK19 [candidate evaluation] **is** done	AR19 candidate evaluator
TK20 academic progress evaluation	PK20 [academic progress evaluation] **is** done	AR20 academic progress evaluator
TK21 diploma control	PK21 diploma control **for** [offering period] **is** done	AR21 diploma controller
TK22 diploma allocation	PK22 **the** diploma allocation **of** [student course registration] **is** done	AR22 diploma allocator

We now discuss the main constructs of DEMOSL 4.5 [27], as represented in Fig. 4, using **bold** style to indicate the type of construct from DEMOSL 4.5 and *italics* when referring to an instance of the construct.

Since it may not be possible to analyze all the operations at an enterprise, Dietz and Mulder [5] suggest that a Scope of Interest (SoI) is explicitly stated. The SoI for our college demonstration case, is *some operations at a college*, e.g., operations where students register for a course and industry partners become sponsors of projects.

One of the key concepts of the CSD is the **elementary transactor role** that resembles a white diamond-disc, combined with a quadrilateral. Three variations of an **elementary transactor role** exist: (1) elementary, (2) self-initiating elementary, and (3) environmental elementary. For each **elementary transactor role**, the TPT **(**Table 2) displays an associated **transaction kind**, **product kind** and **executor role.**

The TPT indicates that the **transactor role** *supervisor allocator (TAR01)* consists of an **executor role** *supervisor allocator (AR01)* and **transaction kind** *supervisor allocation (TK01)* to produce a **product kind** *[supervisor allocation] is done (PK01).* Given the SoI, Fig. 4 indicates that one **environmental or external composite transactor role** exists, i.e., the grey-shaded thick-bordered construct, *student (CTAR02*). Multiple **transactor roles** are linked via **initiator links**. As an example, *supervisor allocator (TAR01)* is initiated (via an **initiator link**) by the **environmental or external composite transactor role** *student (CTAR02*). The default cardinality range for an initiation link is one (1..1), as indicated by Dietz and Mulder [5]. It is also possible that a **transactor role** may initiate multiple instances of a **transaction kind**. As an example, in Fig. 4 the *project controller (TAR05)* initiates zero-to-many (0..*) instances of *project sponsoring (TK02)*.

The SoI determines whether a **transactor role** is represented as white or gray-shaded. The white **elementary transactor role** *supervisor allocator (TAR01)* indicates that the *supervisor allocator* is inside the SoI. Yet, the grey-shaded **environmental or external elementary transactor role** *project sponsor (TAR02)* indicates that the project sponsor is outside the SoI. The **self-activating transactor role** *module revisor (TAR04*) indicates that module revision is initiated and executed by the same transactor role, i.e., the *module revisor (TAR04*).

Since **transactor roles** need to use facts created and stored in transaction banks, an **access link** is used to indicate access to facts. As an example, Fig. 4 indicates that the **transactor role** named *project controller (TAR05)* has reading access via an **access link** to coordination facts and production facts of **transaction kind** *module revision (TK04)*. It is also possible that **transactor 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 **transactor roles** within the SoI (with the SoI defined as *some operations at a college*) need to use facts that are created outside the SoI by two **external multiple original transaction kinds**, namely *MTK01* (with *college facts*) and *MTK02* (with *person facts*).

Facts created as a result of one **transaction kind** may delay (impede) actions of another **transactor role**. The **impediment link** is used to indicate this delaying behavior. As an example, Fig. 4 includes an **impediment link** (a dotted arrow-line) indicating that acts/facts produced via the **transaction kind** *project sponsoring (TK02)* impedes the **transactor role** *ip clearer (TAR03).* The Process Model, discussed in section 4.1 (Fig. 3) could be used to explain the impediment further, i.e. indicating that an instance of *project sponsoring (TK02)* has to reach the *declare* state before an instance if *ip clearance (TK03)* can be r*equested*.

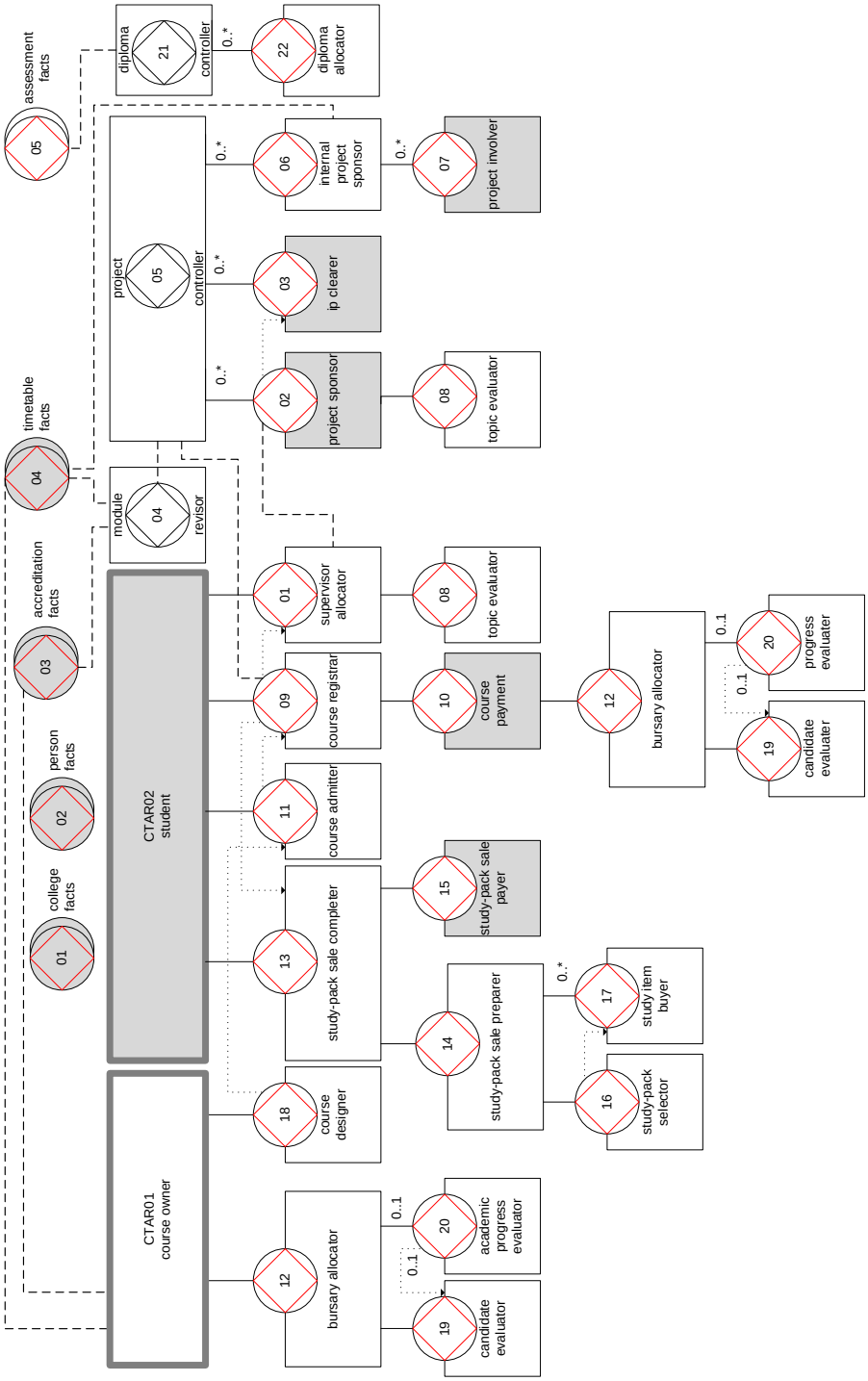

Fig. 4. The CSD for the college case

DEMOSL 4.5 [27] allows to consolidate some **elementary actor roles** within a **composite actor role**. As an example, Fig. 4 indicates that the *course owner (CTAR01)* is represented as a **composite actor role**, since the course owner is within the defined SoI, but may be performing multiple TKs that are not shown explicitly in the diagram. Another construct that is new in DEMOSL 4.5 [27] is the **multiple original transaction kind**, shown as a white double-disc-diamond shape, exemplified by MTK05, named *assessment facts*. MTK05 indicates that multiple *assessment facts* are created as original facts within the SoI.

5. Main Requirements for Transformation Specifications

Evaluating previous TK-BPMN transformations, based on DEMOSL 3.7 [19], we abstracted key requirements to guide the TK-BPMN transformation specifications based on DEMOSL 4.5. The transformation specifications should:

1. Ensure that *appropriate* concepts from BPMN are identified that are conceptually closely related to the DEMO counterpart.
2. Render BPMN CDs to *hide complexity* related to the transaction pattern in a *consistent way*.
3. Render BPMN CDs to ensure easy distinction between *parent-part structures*, as is the case with the PSD.
4. Ensure that the transformed models enable efficient/smooth extension toward execution on dedicated simulation platforms/tools.
5. Ensure that the BPMN CDs are comprehensible for a human being, i.e., it should be possible to generate BPMN CDs with reduced clutter/noise.
6. Accommodate the *standard* pattern for the default TK-BPMN transformation. Since interaction between parent-part structures may initiate a revocation pattern, all four revocation patterns should be accommodated.
7. Allow for extension points to incorporate detailed action rules that are stipulated in DEMO's Action Model.
8. Be comprehensive by means of addressing all possible transformation scenarios when an end-user selects a particular TK for transformation.

In terms of the eighth requirement, we have generated nine transformation scenarios that we present and validate using the college demonstration case that was presented in Section 4.

6. Transformation Scenarios

The nine transformation scenarios as our main contribution, are presented in Section 6.1 and validated in Section 6.2.

6.1. Proposition of Nine Scenarios

For each scenario, an end user selects a single TK that needs to be transformed into BPMN CDs. We identified three key differentiators to distinguish between scenarios, indicated as shaded headings in Table 3. A TK selected for TK-BPMN transformation: (1) is initiated by an actor role that is *not a self-initiating*, (2) *is self-initiating*, and (3) *has part(s)*.

Table 3. TK-to-BPMN transformation scenarios

TK init. by 1 vs * ARs	**TK is self-init.**	**TK has part(s) (0, 1 vs *)**	**Example from Fig. 4 CTAR-initiated**	**Example from Fig. 4 TAR-initiated**
Scenario 1: TK is initiated by 1 AR *AND* has 0 parts				
1	-	0	TK18	TK03, TK07, TK11, TK15, TK16, TK17, TK19, TK20, TK22
Scenario 2: TK is initiated by 1 AR *AND* has 1 part				
1	-	1	TK01, TK9	TK02, TK06, TK10
Scenario 3: TK is initiated by 1 AR *AND* has * parts				
1	-	*	TK13	TK14
Scenario 4: TK is initiated by * ARs *AND* has 0 parts				
*	-	0		TK08
Scenario 5: TK is initiated by * ARs *AND* has 1 part				
*		1		TK12 (when TK20 is removed)
Scenario 6: TK is initiated by * ARs *AND* has * parts				
*	-	*	TK12	TK12
Scenario 7: TK is self-initiating *AND* has 0 parts				
	x	0		TK04
Scenario 8: TK is self-initiating *AND* has 1 part				
	x	1		TK21
Scenario 9: TK is self-initiating *AND* has * parts				
	x	*		TK05

In accordance with Table 3, some distinctions are not required to differentiate between scenarios due to the following reasons:

1. TK-BPMN transformations are sensitive for the type of initiator. Hence, we classify the examples extracted from Fig. 4 according to CTAR (includes *composite* actor role and environmental *composite* actor role), and TAR. The reason is that the CTAR will be modelled as a black box. Yet, when a TK is initiated by a parent TAR, the transformed BPMN should represent the detailed interactions between the parent-part TKs.
2. Nested scenarios are possible during a TK-BPMN transformation, e.g., TK10 conforms to Scenario 2, but its part (i.e., TK12) conforms to Scenario 6.

The nine scenarios in Table 3 highlight another interesting fact, namely that TK12 is classified as an instance of *Scenario 6*, but the TK is initiated by different types of initiators, CTAR-initiated, as well as TAR-initiated. The implication is that when a modeler selects TK12 for transformation, both initiators should be modelled, but in different ways. The CTAR will be modelled as a black box, whereas the TAR will be modelled as a white box.

In Fig. 5 we highlight the nine scenarios that are demonstrated in the college case. For simplicity, we have only indicated one or two examples per scenario on Fig. 5, differentiating between CTAR-initiated and TAR-initiated. Table 3 indicates that the college case offers multiple examples for some of the scenarios.

Addressing the four problems discussed in Section 4.2, the new transformation specifications should:

1. Determine, based on the modeler's selection of parent-part interactions, the valid *acts* and *facts* that need to be included in the BPMN CD, allowing for different initiation scenarios for the parent TK.
2. Provide for multiple *response links* and *wait links* between parent-part structures, incorporating the sequence of acts and facts as indicated in the *complete* transaction pattern.
3. Provide *interimpediment relationships* that exist between TKs, even if the TKs do not form part of parent-part structures.

6.2. Scenario Validation

We have already created a fictitious college case to demonstrate the comprehensive set of nine transformation scenarios (see Fig. 4) and the sub-scenarios within the main scenarios (demonstrated in Fig. 3). For further validation, we used the five cases presented in [5] to validate the nine main transformation scenarios, as indicated in Table 4.

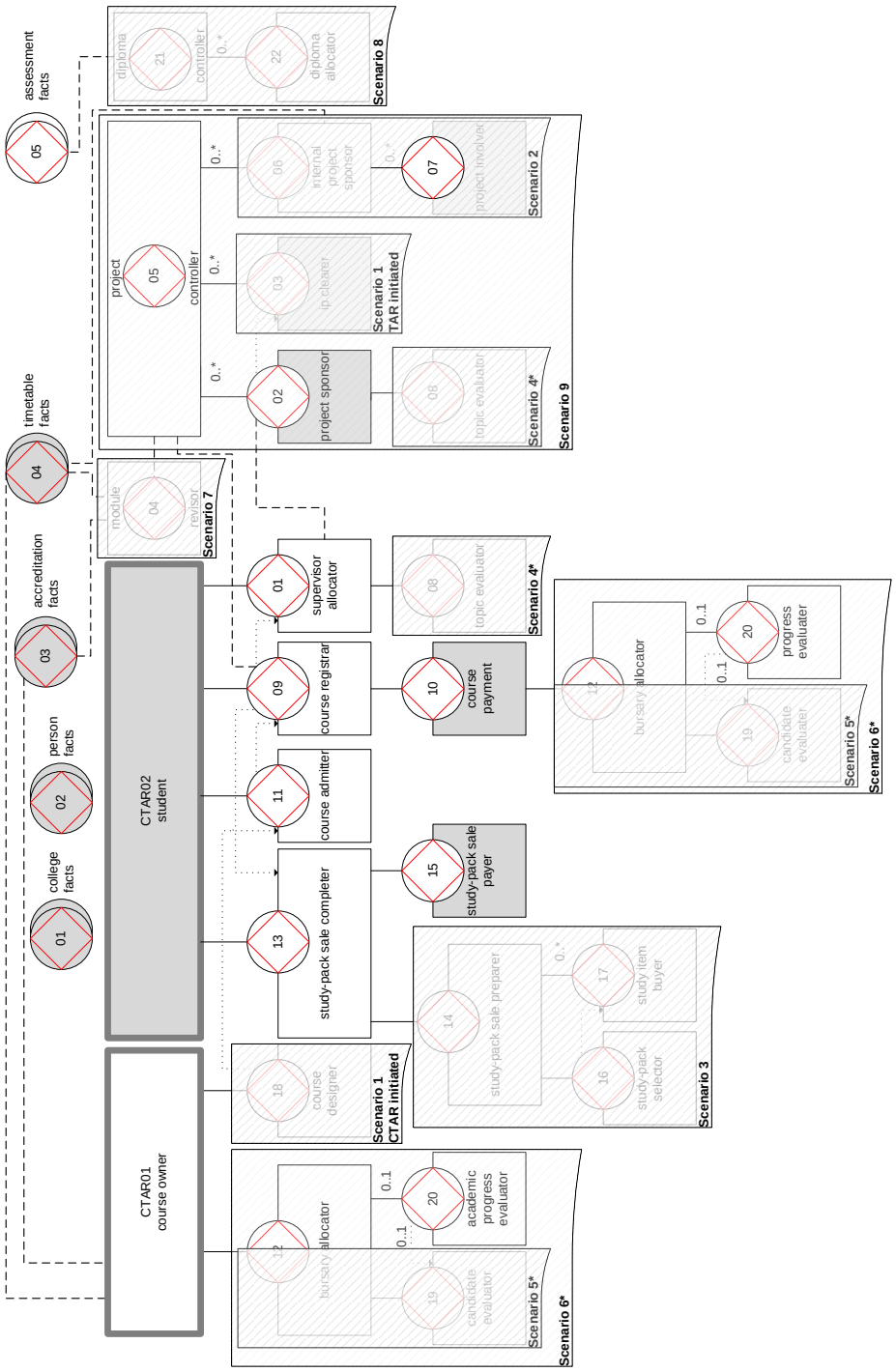

Fig. 5. Nine Scenarios identified in the college case

Table 4. Validating the nine scenarios using cases from [5]

Scenario	TK init. by 1 vs * ARs	TK is self - init.	TK has part(s) (0, 1 vs *)	Case Pizz-eria	Case RAC	Case Lib-rary	Case Poli-Gon	Case GloLog
1	1	-	0	TK02, TK03, TK04	TK04, TK02, TK03, TK04	-	TK02, TK03, TK04	TK10, TK03, TK17, TK07, TK04, TK05, TK06, TK16, TK08, TK09
2	1	-	1	-	-	TK01, TK03	TK01	TK01
3	1	-	*	TK01	TK01		-	TK02, TK14, TK15
4	*	-	0	-	-	TK02, TK04	-	-
5	*		1	-	-	-	-	-
6	*	-	*	-	-	-	-	-
7		x	0	-	-	-	-	-
8		x	1	-	TK07	TK05	-	TK11, TK12, TK13
9		x	*	-	-	-	-	-

Our validation results in Table 4 indicate that the nine transformation scenarios are comprehensive to address all five cases presented in [5]. In addition, we highlighted four transformation scenarios that are currently under-represented by existing cases, namely scenario's 5, 6, 7 and 9. Since the fictitious college case addresses all nine scenarios, as indicated in Table 3, we believe that the college case could also be used as a valid case for developing the TK-BPMN transformation specifications that should be based on DEMOSL 4.5.

7. Conclusion and Future Research

The DEMO approach represents the organization design of an enterprise in four linguistically based, semantically sound aspect models. The Business Process Model and Notation (BPMN) on the other hand enables more flexibility during modelling and benefits from wide adoption in industry, the execution of processes and the availability of proper tooling. A transformation of DEMO models into BPMN models is thus desirable to enable both, the semantic sound foundation of the DEMO models and the wide adoption and execution possibilities of the de-facto industry standard BPMN.

Using Design Science Research, this paper presented a list of main requirements for developing model transformation specifications to transform concepts represented on

the coordination structure diagram (CSD), associated transactor product table (TPT) and the process structure diagram (PSD) of DEMO to concepts of the BPMN Collaboration Diagrams (CDs). The main contribution of this article is the identification and presentation of nine generic transformation scenarios that we validated by applying them to multiple demonstration cases that already exist in literature. We thereby showed not only the comprehensiveness of the transformation scenarios but also the underrepresentation of four scenarios in current DEMO cases.

Since the fictitious college case addresses all nine scenarios, we believe that the college case could be used as a valid case for developing detailed TK-BPMN transformation specifications that are also based on DEMOSL 4.5. With this paper we thus, secondly, contribute a comprehensive DEMO case that the enterprise engineering community can further refine, validate, and use. We believe that the nine scenarios would not only by useful for TK-BPMN transformations, but also for transformations from DEMO coordination-related models to other coordination-related languages.

The eight requirements for TK-BPMN transformation specifications, combined with the nine transformation scenarios, as well as three additional parent-part-interaction requirements, will guide our future work.

References

1 Bork, D., Buchmann, R., Karagiannis, D.: Preserving multi-view consistency in diagrammatic knowledge representation. . In: Int. Conf. on Knowl., Sci., Eng. and Manag., pp. 177-182. Springer, Cham (2015). 10.1007/978-3-319-25159-2_16

2 Broy, M., Freilkas, M., Hermannsdoerfer, M., Merenda, S., Ratiu, D.: Seamless model-based development: From isolated tools to integrated model engineering environments. Proceedings of the IEEE, **98**(4), 526-545 (2010)

3 Awadid, A., Nurcan, S.: Consistency requirements in business process modeling: a thorough overview. Software & Systems Modeling, **18**(2), 1097-1115 (2019). 10.1007/s10270-017-0629-2

4 Cicchetti, A., Ciccozzi, F., Pierantonio, A.: Multi-view approaches for software and system modelling: a systematic literature review. Softw. and Syst. Model., **18**(6), 3207-3233 (2019). 10.1007/s10270-018-00713-w

5 Dietz, J. L. G., Mulder, H., B, F.: Enterprise ontology: A human-centric approach to understanding the essence of organisation. Springer International Publishing AG (2020)

6 Graumann, C. F., Kallmeyer, W.: Perspective and perspectivation in discourse. John Benjamins Publishing Company, Amsterdam (2002)

7 Dumas, M., La Rosa, M., Mendling, J., Reijers, H. A.: Fundamentals of business process management (2nd ed.). Springer, Berlin, Germany (2018)

8 Dietz, J. L. G.: Enterprise ontology. Springer, Berlin (2006)

9 Silver, B.: BPMN method and style, second edition, with BPMN implementer's guide. Cody-Cassidy Press, USA, Aptos (2011)

10 De Vries, M.: DEMO and the story-card method: Requirements elicitation for agile software development at scale. In: Buchmann, R., Kirikova, M. (eds.) 11th IFIP WG 8.1 Working Conference on the Practice of Enterprise Modelling. Springer (2018). https://doi.org/10.1007/978-3-030-02302-7

11 Grigorova, K., Mironov, K.: Comparison of business process modeling standards. Int. J. of Eng. Sci. & Manag. Res., **1**(3), 1-8 (2014). 10.1109/SOLI.2006.328910

12 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.) LNBIP. Vol. 34, pp. 115-129. Springer-Verlag, Berlin Heidelberg (2009). 10.1007/978-3-642-01915-9_9
13 Barjis, J.: Enterprise modeling and simulation within enterprise engineering. J. of Enterp. Transform., **1**(3), 185-207 (2011). doi.org/10.1080/19488289.2011.601399
14 Figueira, C., Aveiro, D.: A new action rule syntax for DEMO models based automatic worKflow process generation (DEMOBAKER). In: Aveiro, D., Tribolet, J., Gouveia, D. (eds.) Advances in Enterprise Engineering VIII, LNBIP 174. Springer, Switzerland (2014). doi.org/10.1007/978-3-319-06505-2_4
15 Rodrigues, D. P. (2017). Modelling business processes in BPMN inspired by the DEMO principles. Técnico Lisboa
16 Gray, T., Bork, D., De Vries, M.: A new DEMO modelling tool that facilitates model transformations. In: Nurcan, S., Reinhardt-Berger, I., Soffer, P., Zdravkovic, J. (eds.) Enterprise, Business-Process and Information Systems Modeling. Vol. LNBIP 387, pp. 359-374. Springer Nature Switzerland, Cham, Switzerland (2020). DOI: 978-3-030-49418-6_25
17 Bork, D., Buchmann, R. A., Karagiannis, D., Lee, M., Miron, E.-T.: An open platform for modeling method conceptualisation: The OMiLAB digital ecosystem. Commun. of the AIS, **44**(32), 673-697 (2019). doi.org/10.17705/1CAIS.04432
18 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). DOI: 10.1201/b12295-17
19 Gray, T., De Vries, M.: Empirical evaluation of a new DEMO modelling tool that facilitates model transformations. In: Dobbie, G., Frank, U., Kappel, G., Liddle, S. W., Mayr, H. C. (eds.) 39th International Conference on Conceptual Modeling. (2020 - In press).
20 Hoogervorst, J. A. P.: Practicing enterprise governance and enterprise engineering - applying the employee-centric theory of organization. Springer, Berlin Heidelberg (2018)
21 De Vries, M.: Towards consistent demarcation of enterprise design domains. In: De Cesare, S., Frank, U. (eds.) Advances in Conceptual Modeling. pp. 91-100. Springer, Switzerland (2017). doi.org/10.1007/978-3-319-70625-2_9
22 Decosse, C., Molnar, W. A., Proper, H. A.: What does DEMO do? A qualitative analysis about DEMO in practice: Founders, modellers and beneficiearies. In: Aveiro, D., Tribolet, J., Gouveia, D. (eds.) Advances in Enterprise Engineering VIII. Springer, Portugal (2014). doi.org/10.1007/978-3-319-06505-2_2
23 Peffers, K., Tuunanen, T., Rothenberger, M., Chatterjee, S.: A design science research methodology for information systems research. J. of MIS, **24**(3), 45-77 (2008). doi.org/10.2753/MIS0742-1222240302
24 Guizzardi, G., Wagner, G.: Can BPMN be used for making simulation models? In: Barjis, J., Eldabi, T., Gupta, A. (eds.) Lecture notes in business information processing. Vol. 88, pp. 100-115. Springer, Heidelberg (2011). 10.1007/978-3-642-24175-8 8
25 Mraz, 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.) LNBIP 284. pp. 85-98. Springer International Publishing, Cham, Switzerland (2017). DOI: 10.1007/978-3-319-57955-9_7
26 Perinforma, A. P. C.: The essence of organisation (3rd ed.). Sapio, www.sapio.nl (2017)
27 Dietz, J. L. G.: The DEMO Specification Language Version 4.5. (2020)