Transforming Transaction Models into ArchiMate

Sybren de Kinderen[1], Khaled Gaaloul[1], and H.A. (Erik) Proper[1,2]

[1] CRP Henri Tudor
L-1855 Luxembourg-Kirchberg, Luxembourg
sybren.dekinderen, khaled.gaaloul, erik.proper@tudor.lu
[2] ICIS, Radboud University Nijmegen
P. O. BOX 9010 6500, GL Nijmegen, The Netherlands

Abstract. ArchiMate, a language for modelling an organisation from a holistic perspective, lacks guidelines and techniques for exploring each of its perspectives in depth. To address this issue, we propose to use the DEMO modelling technique and toolset as a front-end for ArchiMate. In particular, DEMO adds to ArchiMate a conceptual clarity, as well as tools and techniques for modelling business processes.
Specifically, in this paper we contribute a formal model transformation from DEMO to ArchiMate, and show how this model transformation can be used to transform DEMO models into ArchiMate models. Our model transformation approach is illustrated by a fictitious but realistic case study from the insurance domain.
Keywords: ArchiMate, DEMO, meta model, model transformation

1 Introduction

ArchiMate, is an Open Group standard [1, 2] for the modelling of enterprise architectures[3]. Being designed as a general purpose modelling language for enterprise architecture [1], it allows architects to model an enterprise from a holistic perspective, showing amongst others, an organization's products and services, how these products and services are realized/delivered by business processes, and how in turn these processes are supported by information systems and their underlying IT infrastructure. This holistic perspective on an enterprise helps to guide change processes [3], provides insight into cost structures [4], and more [1].

Because of the inherent holistic nature, ArchiMate lacks specificity on how to model the different perspectives *in-depth*. For example, ArchiMate lacks guidelines for process modelling, and lacks expressivity for modelling an enterprise from a value exchange perspective [5]. Moreover, as claimed by [6] ArchiMate lacks conceptual clarity and precision. This "lack" is, however, a direct consequence of the coarse-grained, and holistic, nature of ArchiMate. In that sense this *freedom of interpretation* has been designed into the language on purpose [1]. Nevertheless, as a result, different modellers do indeed create different models.

[3] http://www.opengroup.org/archimate/

To address the above issues, it has already been suggested that ArchiMate could benefit from the integration with a method such as DEMO to provide it with a more explicit way of working, supporting architects in the creation of models [6]. In this paper, we focus on bridging between DEMO and ArchiMate.

DEMO, short for Design and Engineering Methodology for Organizations, is a method comprising of a comprehensive set of conceptual modelling techniques, in combination with a theory based a way of thinking and associated way of working, focused on modelling/analysing/designing the *essential* aspects of an organization [7, 8]. DEMO uses the word *essential* here to refer to the implementation-independent aspects of an organization. As such, DEMO aims to abstracts away from implementation-specific details, such as the information systems present in business collaboration. Linking DEMO and ArchiMate would enable architects to use the semantically rich way of thinking of DEMO to create ArchiMate models. These models would then primarily be ArchiMate models providing an essential view of the business processes (business layer) and the information processing (application layer) in the enterprise. Further benefits of linking DEMO and ArchiMate are discussed in [9].

The core contribution of this paper is two-fold: (1) a formal mapping of the meta models underlying DEMO and ArchiMate, accompanied by a rationale of why such a mapping is beneficial (2) a systematic application of the DEMO and ArchiMate meta models to map a model created in DEMO to a model of an enterprise architecture in ArchiMate. We use a running example of an insurance scenario to illustrate our ideas.

The remainder of this paper is structured as follows. Sect. 2 illustrates, by means of the case of an insurance company, how we intend to use DEMO as a front-end to ArchiMate. Therafter, we show how to formally transform a DEMO model into an ArchiMate model (Sect. 3).

2 Modelling insurance processes in DEMO

2.1 Archinsurance: selling car insurance via insurance brokers

We use the insurance company Archinsurance, as documented in [3, 10], as a fictitious but realistic use case. We focus on car insurance, an insurance product that Archinsurance sells via *insurance brokers*. The main reason for selling insurance via brokers is to reduce the risk of *adverse risk profiles* [11], incomplete or faulty risk profiles of customers that lead insurance companies to sell inappropriate insurance packages.

2.2 Using DEMO transaction patterns for process modelling

We use DEMO to model the sale of car insurance by Archinsurance. The DEMO meta model is depicted in Fig. 2, with an instantiation of this model, called the DEMO process model, in Fig. 1. Chief to the creation of this process model are DEMO *transaction patterns*.

A DEMO transaction pattern is a process-based pattern of (instantiations of) DEMO meta model concepts, showing the sequence of acts that *always* needs to be executed to realise a social interaction between two actors (this interaction being called a DEMO 'transaction'). So, here we see DEMO's emphasis on the essential aspect of an organisation, as mentioned in the introduction: no matter what the domain, if we perceive of an organisation as a social entity, then we see a pattern of generic acts that *always* occurs in carrying out a transaction [8]. So, for example, one actor always has to initiate a transaction by performing the act 'request' (which in the Archinsurance case may translate to the act 'Apply for insurance' as carried out by a customer), while another actor has to always perform the 'execute' act in order to produce the good or service that the initiating actor is interested in (see Fig. 1). In the Archinsurance case, this may translate to the act 'Find matching package' which, as mentioned before, is executed by the insurance broker. Such patterns are particularly useful as they guide us in creating process models, whereas a language on its own, such as ArchiMate, cannot.

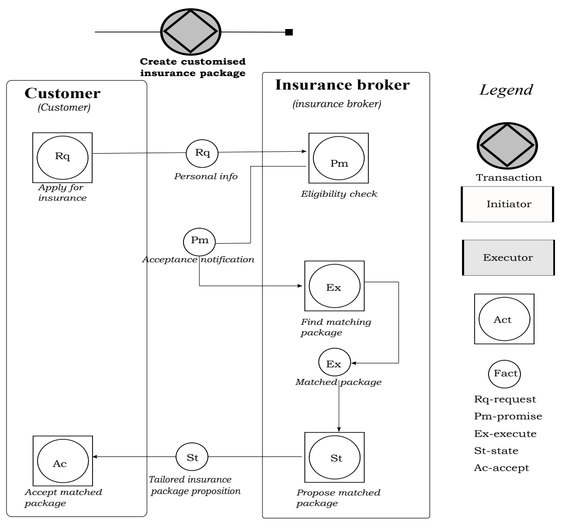

Fig. 1. An excerpt of the DEMO Business process model

3 Translating DEMO process models to ArchiMate

We now show how to create an ArchiMate enterprise architecture model, using the DEMO process model as a baseline. To this end, we first introduce the used mapping technique (in Sect. 3.1), and subsequently apply this technique to transform a DEMO process model to an ArchiMate model (in Sect. 3.2).

3.1 The used mapping technique

For mapping DEMO to ArchiMate, we use the meta model mapping technique described in [12] where authors distinguish different types of mappings, the most relevant for our work being (1) *class-to-class mappings*, which relates a concept from meta model A to a concept from meta model B (e.g., a 'Subject' from DEMO relates to an 'Actor' from ArchiMate). And (2) *relation-to-relation mappings*, which relates concept relationships from meta model A with concept relationships from meta model B (e.g., 'performs_role' between the concepts Subject and Actor from DEMO relates to the ArchiMate relation 'assigned_to' between the concepts Actor and Business role).

3.2 Mapping the DEMO meta model to the ArchiMate meta model

Now we translate a DEMO process model for Archinsurance into an ArchiMate business layer model. We do this in two main steps: (1) Translate the concepts from a DEMO process model to an ArchiMate process model, which we can do given that, looking at the concept definitions, the holistic ArchiMate language subsumes DEMO's social perspective, (2) Define a (partial) enterprise architecture model from a business perspective that focuses on the DEMO process model. Here, we construct an ArchiMate model from the mapped DEMO concepts. As we now actually construct an ArchiMate model, we take here into consideration (a) the difference in abstraction level between DEMO and ArchiMate, and (b) additional ArchiMate constructs not present in DEMO, for example for depicting an IT perspective on the organisation at hand.

Step 1: Horizontal integration via meta model mapping The first step will apply our mapping between the DEMO meta model and the ArchiMate business layer meta model (see Fig. 2). Here, we make a mapping on a purely *horizontal* level (cf. [12]), meaning that we consider only differences between aspects modelled in DEMO and ArchiMate *on the same abstraction level.* In doing so, we apply the DEMO - ArchiMate meta model mapping from Fig. 2, and the corresponding rationale of our meta model mapping (i.e., Table 1). In Fig. 2, we define a specialisation relation between the mapped concepts from DEMO to ArchiMate concepts.

For Archinsurance, we apply this mapping as follows. For reference, see the Archinsurance ArchiMate model in Fig. 3, and the Archinsurance DEMO process model in Fig. 1. For instance, as explained in Table 1 *subjects from DEMO map to business actors in ArchiMate.* For Archinsurance, we thus map the subject 'customer' to the business actor 'customer' in ArchiMate. Due to space restrictions, we will not exemplify with our scenario the rest of concept mappings explained in Table 1.

In addition, we perform relation-to-relation mappings between DEMO and ArchiMate (see Table 2). As such, we map *the relation (Subject)performs_role(Actor) from DEMO to the relation (Business actor)assigned_to(Business role) from*

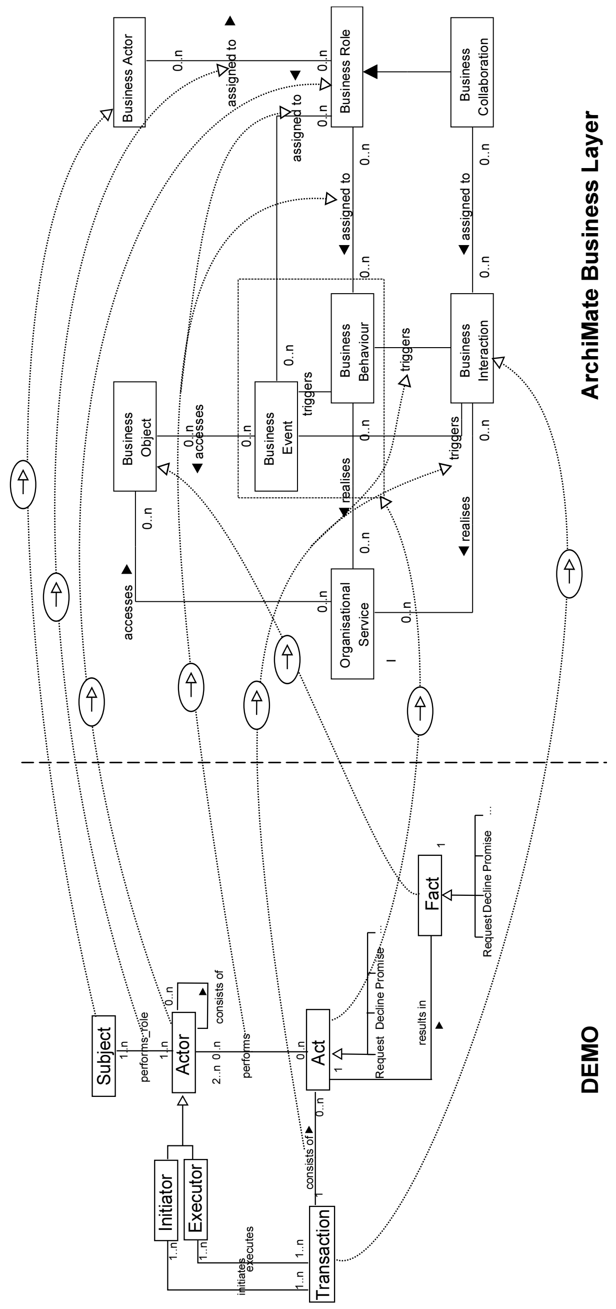

Fig. 2. Mapping of DEMO and ArchiMate meta models

ArchiMate. For example, in both DEMO and ArchiMate, the department 'insurance broker' performs the role of 'insurance broker'. The relation-to-relation mappings are explained in Table 2.

Table 1. DEMO - ArchiMate meta model concept mapping relations

DEMO - ArchiMate Concepts	Mapping rationale
Actor Business role	In DEMO, an actor refers to a social role played by a subject in an organisation. Such a social role corresponds to the definition of a business role in ArchiMate where roles are typically used to distinguish responsibilities.
Subject Business actor	A DEMO subject is an organisational entity - person, department or otherwise - that can fulfil an organisational role. This corresponds to a business actor in ArchiMate, which is an organisational entity that performs some behaviour (cf. [3]), thus it can also fulfil a role.
Act Business behaviour/event	An act is performed by a subject in a social role. Its scope is about contribution/coordination for services. In the ArchiMate context, it corresponds to the realisation of an organisational service via a business process or a function (business behaviour) or a business event (e.g., an external request).
Transaction Business interaction	For DEMO transactions, the initiation and execution are performed by different actors. This emphasises the interaction aspect that we can find in ArchiMate, where a business interaction is carried out by more than one actor.
Fact Business object	A fact is any object that results from performing an act. In ArchiMate, this corresponds to a business object, which 'represent the important concepts in which the business thinks about a domain' [3].

Step 2: Vertical integration: defining an appropriate abstraction level in ArchiMate The second step consists of defining an enterprise architecture model using ArchiMate to represent a DEMO process model.

First, in addition to the horizontal differences in Step 1, we now consider also the *vertical* differences between DEMO and ArchiMate. This means that we remove from the ArchiMate model any elements that are too detailed for depicting a holistic perspective on the organisation at hand. For example: for the Archinsurance case, we thus remove the business object 'acceptance notification' (which ArchiMate inherits from the DEMO process model in Fig. 1), since they are too detailed for the high-level model overview provided by ArchiMate.

Second, we supplement the model elements inherited from DEMO with ArchiMate constructs. This we do to fully express a holistic perspective on the organisation at hand, most prominently in terms of the supporting IT infrastructure. For example, as we can see in Fig. 3, for Archinsurance we model that

Table 2. DEMO - ArchiMate meta model relations mapping

DEMO - ArchiMate relation	Mapping rationale
performs_role assigned_to	In both DEMO and ArchiMate, one relates a real world entity (e.g., Archinsurance) to a role played by that entity (e.g., the role of insurer in the case of Archinsurance).
consists_of triggers	As transactions map to business interactions, and acts map to business events and business behaviour, the relation 'consists_of' between transactions and acts in DEMO maps logically to the relation 'triggers' between business interactions and business events/business behaviour in ArchiMate.
performs assigned_to	While both use different nomenclature, in both DEMO and ArchiMate, a role - not the real-world entity behind it - carries out acts.

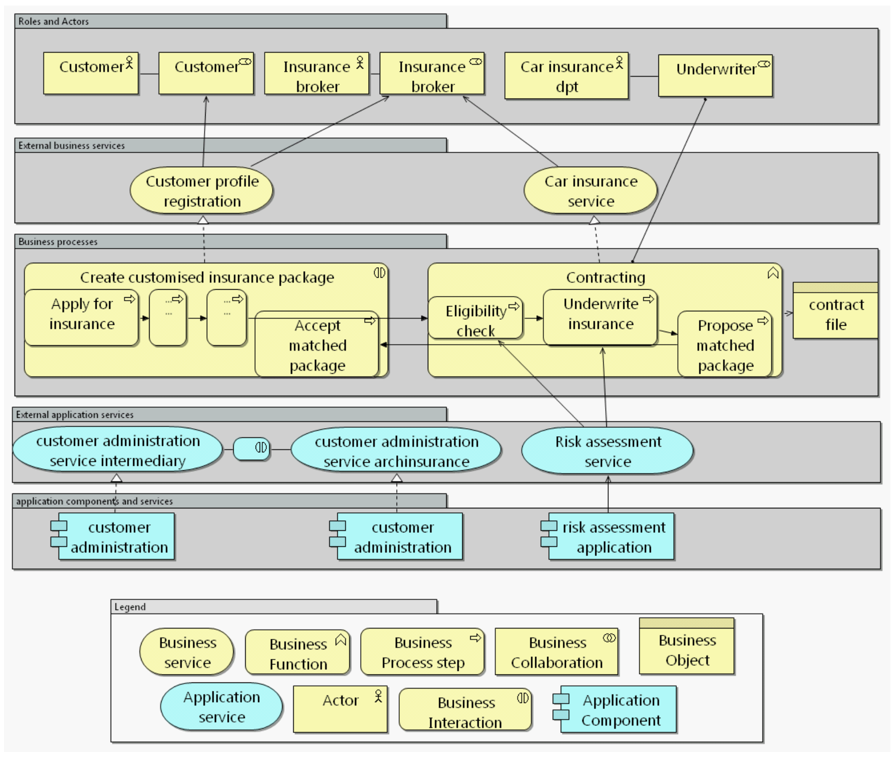

Fig. 3. (Partial) enterprise architecture model based on DEMO process

the business process activities 'eligibility check' and 'underwrite insurance' are supported by a risk assessment application, and that the business collaboration 'create customized insurance package' is supported by administrative applications from both the insurance broker and Archinsurance.

4 Conclusion and Future Work

In this paper, we used DEMO as a front end for ArchiMate to help creating enterprise architecture models. Using a fictitious case from the insurance domain, we introduced a computationally supported mapping between DEMO and ArchiMate, and showed how this mapping can be applied to translate a DEMO model into an ArchiMate model.

For further research, we will look into enriching ArchiMate itself with other techniques, for example to add expressivity from a value perspective. This naturally introduces a number of research challenges, such as how to balance model integration - changing ArchiMate itself - with model transformation - leaving a concern to a specific technique, and import results into ArchiMate.

References

1. M.M. Lankhorst et al. *Enterprise Architecture at Work: Modelling, Communication and Analysis.* Springer, Berlin, Germany, 2005.
2. M.-E. Iacob, H. Jonkers, M.M. Lankhorst, and H.A. Proper. *ArchiMate 1.0 Specification.* The Open Group, 2009.
3. H. Jonkers, M.M. Lankhorst, R. van Buuren, S.J.B.A. Hoppenbrouwers, M. Bonsangue, and L. Van der Torre. Concepts for Modeling Enterprise Architectures. *International Journal of Cooperative Information Systems*, 13(3):257–288, 2004.
4. R. van Buuren, J. Gordijn, and W. Janssen. Business case modelling for e-services. In *18 th Bled eConference eIntegration in Action*, 2005.
5. V. Pijpers, J. Gordijn, and H. Akkermans. e3alignment: Exploring inter-organizational alignment in networked value constellations. *International Journal of Computer Science & Applications*, page 59, 2009.
6. R. Ettema and J.L.G. Dietz. Archimate and demo–mates to date? *Advances in Enterprise Engineering III*, pages 172–186, 2009.
7. J.L.G. Dietz. *Enterprise ontology: theory and methodology.* Springer Verlag, 2006.
8. J.L.G. Dietz. The deep structure of business processes. *Communications of the ACM*, 49(5):58–64, 2006.
9. S. de Kinderen, K. Gaaloul, and E. Proper. On transforming demo models to archimate. In *To appear in the Proceedings of the 2012 EMMSAD/Eurosymposium workshop, Gdansk, Poland.* Springer Verlag, 2012.
10. M.M. Lankhorst, H.A. Proper, and H. Jonkers. The Architecture of the ArchiMate Language. *Enterprise, Business-Process and Information Systems Modeling*, pages 367–380, 2009.
11. J.D. Cummins and N.A. Doherty. The economics of insurance intermediaries. *The Journal of Risk and Insurance*, 73(3):359–396, 2006.
12. S. Zivkovic, H. Kuhn, and D. Karagiannis. Facilitate modelling using method integration: An approach using mappings and integration rules. 2007.