ArchiMate and DEMO – Mates to Date?

Roland Ettema[1] and Jan L.G. Dietz[2]

[1] Logica Consulting Netherlands
roland.ettema@gmail.com
[2] Delft University of Technology, The Netherlands
j.l.g.dietz@tudelft.nl

Abstract. ArchiMate is an approach to modeling the architecture of enterprises. In the corresponding architecture framework, three enterprise layers are distinguished: business, application and technology. Although ArchiMate is broadly applied in practice, its semantics appears to be undefined. DEMO is a methodology for enterprise engineering that is facing a rapidly growing acceptance. It is firmly rooted in a sound and appropriate theoretical basis. DEMO also distinguishes between three enterprise layers: ontological, infological and datalogical. This paper reports on a theoretical and practical comparative evaluation of ArchiMate and DEMO. Only the business layer of ArchiMate and the ontological layer of DEMO are considered. Three conclusions are drawn. First, the two approaches are hardly comparable since ArchiMate belongs to the second and DEMO to the third wave of approaches. Second, the business layer of ArchiMate corresponds to all three layers of DEMO, without a possibility to distinguish between them. Third, ArchiMate could benefit from adopting DEMO as its front-end approach, thereby enforcing the rigorously defined semantics of DEMO on the Archimate models.

Keywords: ArchiMate, DEMO, Enterprise Engineering, Enterprise Architecture, Enterprise Ontology.

1 Introduction

In the ongoing turmoil of emerging paradigms, methods and techniques in the field of Enterprise Engineering, and the sometimes hot theoretical and practical discussions concerning them, occasionally approaches pop up that survive despite their actual or alleged shortcomings. Two current examples of such approaches, incidentally both originating from The Netherlands, are Archimate and DEMO. We have selected them for investigation and assessment because we think that a combination of the two could be beneficial. ArchiMate is a modeling language for enterprise architecture. The main information source regarding Archimate was [11]; however, since it has become a standard of The Open Group (TOG), the authoritative source is now the description

A. Albani, J. Barjis, and J.L.G. Dietz (Eds.): CIAO!/EOMAS 2009, LNBIP 34, pp. 172 – 186, 2009.

by TOG[1]. DEMO (Design and Engineering Methodology for Organizations) is an enterprise engineering methodology. The authoritative source for DEMO is [4]. This paper reports on the research we have conducted for the sake of articulating and elucidating the differences between Archimate and DEMO, in order to enable a sensible assessment of a possible combination. We have done that in the context of the new discipline of Enterprise Engineering. Although this discipline is certainly not fully established yet, the main characteristics are becoming clear [8]. They are summarized in the Enterprise Engineering Manifesto[2].

One of these characteristics is that Enterprise Engineering is the result of the merging of (the current state of) the Information Systems Sciences and the Organization Sciences. Within the former, three phases or waves can be distinguished in the understanding of the application of Information and Communication Technology (ICT) to enterprises. The first wave started with the introduction of computers in the sixties, and ended in the seventies of the 20th century. The few available approaches at that time focused on the form of information. Applying ICT basically meant replacing a paper document by its electronic equivalent. During the seventies a 'revolution' took place, pioneered by Langefors [10], who suggested to focus on the content of information before bothering about its form. Applying ICT began to mean automating information (i.e. content) needs, regardless the form in which the information is stored or presented. It marks the beginning of the second wave. Around 2000 another 'revolution' started, pioneered by people from the Language-Action Perspective community [6]. This marks the beginning of the third wave. Basing their insights on language philosophy [1, 7, 12], they suggested to recognize the intention of information on top of its content and to focus on this aspect first, before bothering about the content and the form. Examples of intentions are: request, promise, state, and accept. Because the informational notion of intention is closely related to the organizational notions commitment and responsibility, the 'natural' merge became possible of the information system sciences and the organizational sciences into the discipline of enterprise engineering. Since ArchiMate is based on the descriptive notion of architecture [5], we can safely equate the architecture of an enterprise with a conceptual model of its business processes and objects. From the description of Archimate it becomes clear that it is a second wave approach, meaning that it ignores the intention aspect of communication and information. DEMO is clearly a third wave approach. Yet, its scientific foundation is broader. Next to language philosophy, it includes system ontology [2] and world ontology [16].

Another main characteristic of Enterprise Engineering is a profound understanding of the process of system development of any kind, thus also of enterprises. For changing an enterprise, in particular for supporting its operational activities by means of ICT applications, one needs to have and appropriate understanding of the (stable) essence of an enterprise. From the engineering sciences in general it is known that if one wants to change a system, something of it must remain the same. For example, if

[1] www.opengroup.org/archimate
[2] See www.ciao-network.org

one wants to redesign a meeting room, it is important that it remains a meeting room. As another example, if one wants to support or even replace the employees in an accounting department by means of an automated accounting system, the accounting process must essentially stay untouched. So, in general one needs to have an understanding of a thing to be changed at a level just above the level at which the changes take place. If this understanding is lacking, one even cannot evaluate a change sensibly. For a correct understanding of the process of system development, DEMO relies on the Generic System Development Process [5]. Regarding Archimate, we did not find something similar. This reinforces the observation made earlier, that Archimate is only a modeling language, not a methodology.

Next to the theoretical investigation of Archimate and DEMO, we make a practical comparison by applying both to the case Car Import, that is taken from the ArchiMate project deliverable D3.5.1b [9]. Below the original narrative description is presented:

In The Netherlands any imported car is subjected to special kind of taxation called BPM. The business architecture supporting the whole collection process and the interaction of the Dutch Tax Department (Belastingdienst) with the importers and a number of other parties is described below. The importer (private person or car dealer/importer) must announce himself at the customer counter in any of the 30 Customs units in The Netherlands with the imported vehicle, its (provenance) documents, the approval proof of its technical inspection, and possibly with cash for the payment of the BPM tax. The public servant will handle the tax declaration as follows: first he will check all the documents, then he will fill in all the data into a client BPM application (running on a local server) and will calculate the due BPM tax value (using the BPM application and the catalogue value for that particular car). One copy of the BPM form (BPM17 ex 1) will be issued and sent to the administration. Another copy of this form is handed to the importer (BPM17 ex3), together with either the evidence of a cash payment (if the importer is able to pay the BPM amount in cash), or with a bill ("acceptgiro") issued for the due amount (in the case the importer is not able to pay in cash).

At each Customs unit there will be public servants assigned to handle the additional administrative operations regarding all the incoming BPM statements. Once a day, this person will collect all the incoming BPM17 forms. For ones, which were paid in cash, he will issue and authorize another copy of the BPM form (BPM17 ex2). This copy will be sent to RDW ("Rijksdienst voor het Wegverkeer" - the Netherlands Road Transport Department), which keeps the evidence of all registered vehicles in The Netherlands. The first copy of BPM 17 will be then sent to the archive. The forms which are not yet paid, are kept "on hold" until they are paid. The payment administration and the notification service for all incoming payments for these BPM forms is done by a separate department of the Belastingdienst, namely the Tax Collection Departments ("Inning"), which is responsible for the collection of all payments via bank. Once such a notification is received (via the BPM server application) the

administration will prepare, authorize and send the copy of BPM17 ex.2 to RDW, and will permanently archive the ex1 of the BPM17.

The remainder of the paper is organized as follows. Section 2 summarizes the ArchiMate approach, as well as the analysis of the case Car Import with ArchiMate. Section 3 summarizes the DEMO approach, as well as the DEMO analysis of the case. In Section 4 we compare the two approaches, both theoretically and on the basis of the respective analyses of the case Car Import. Section 5 contains some salient conclusions regarding the differences as well as regarding the possible combination of ArchiMate and DEMO.

2 ArchiMate

2.1 Summary of ArchiMate

ArchiMate is a language for modeling enterprise architectures in accordance with a meta model and a conceptual framework of modeling concepts, called the ArchiMate Framework. ArchiMate is based on the descriptive notion of architecture [8], which means that an enterprise architecture in ArchiMate corresponds to a conceptual model of the business processes in the enterprise. The ArchiMate Framework is exhibited in Fig. 1. Three architectural layers are distinguished, called the business layer, the application layer, and the technology layer. The idea behind this division is that the application layer provides services to the business layer, and that the technology layer provides services to the application layer. Moreover, the business layer is said to provide business services to the environment of the enterprise.

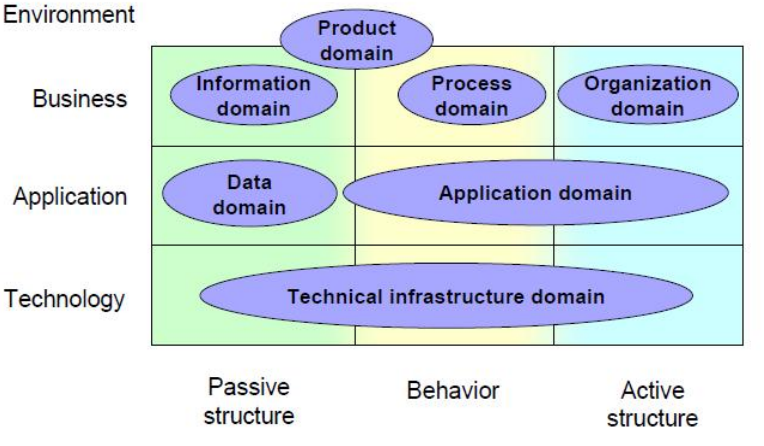

Fig. 1. The ArchiMate Framework

On the horizontal axis, three major aspects are distinguished, called active structure, behavior, and passive structure. The first two refer to generic system aspects, as e.g. identified by Weinberg [15]. The third aspect is related to the discipline of information system engineering. ArchiMate's perspective on organizations is highly influenced by this discipline. The meta model (see Fig. 2) is structured in conformity with the framework of Fig. 1.

The meta model therefore consists of active structural elements, of behavioral elements and of passive structural elements. Fig. 2 shows the meta model for the business layer. The concepts on the right hand side regard the active structure aspect. The concepts in the centre regard the behavioral or dynamic aspect, and the concepts on the left hand side regard the passive aspect.

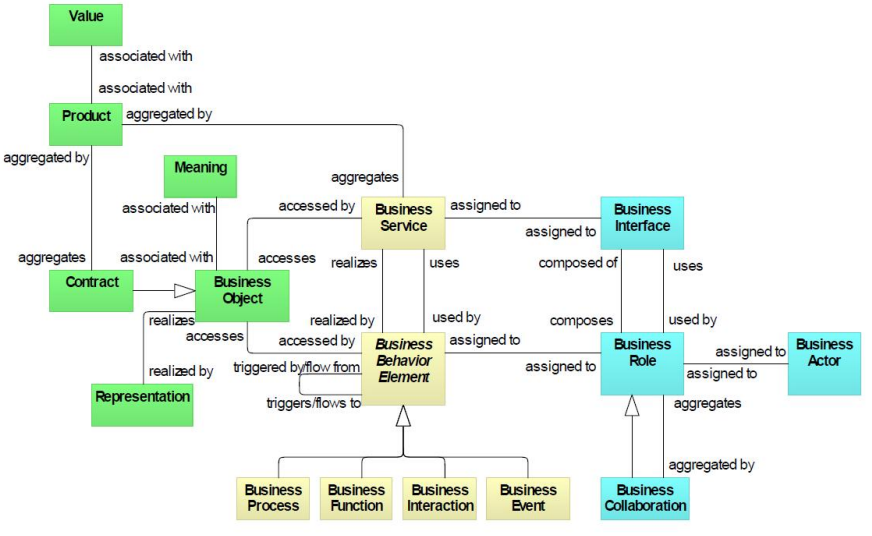

Fig. 2. ArchiMate Meta Model of the business layer

2.2 Analysis of the Case Car Import with ArchiMate

The narrative description of the case Car Import, as presented in section 1, constitutes the starting point for the modeling activity with ArchiMate. The first methodological step is to identify text elements that can be recognized as ArchiMate concepts. The second step is to position these elements within the framework and to determine the relationships that exist between them. The source from which we take the ArchiMate analysis of the case [9] does not provide further details about the modeling activity that has lead to the result as exhibited in Fig. 3. It merely only presents this result.

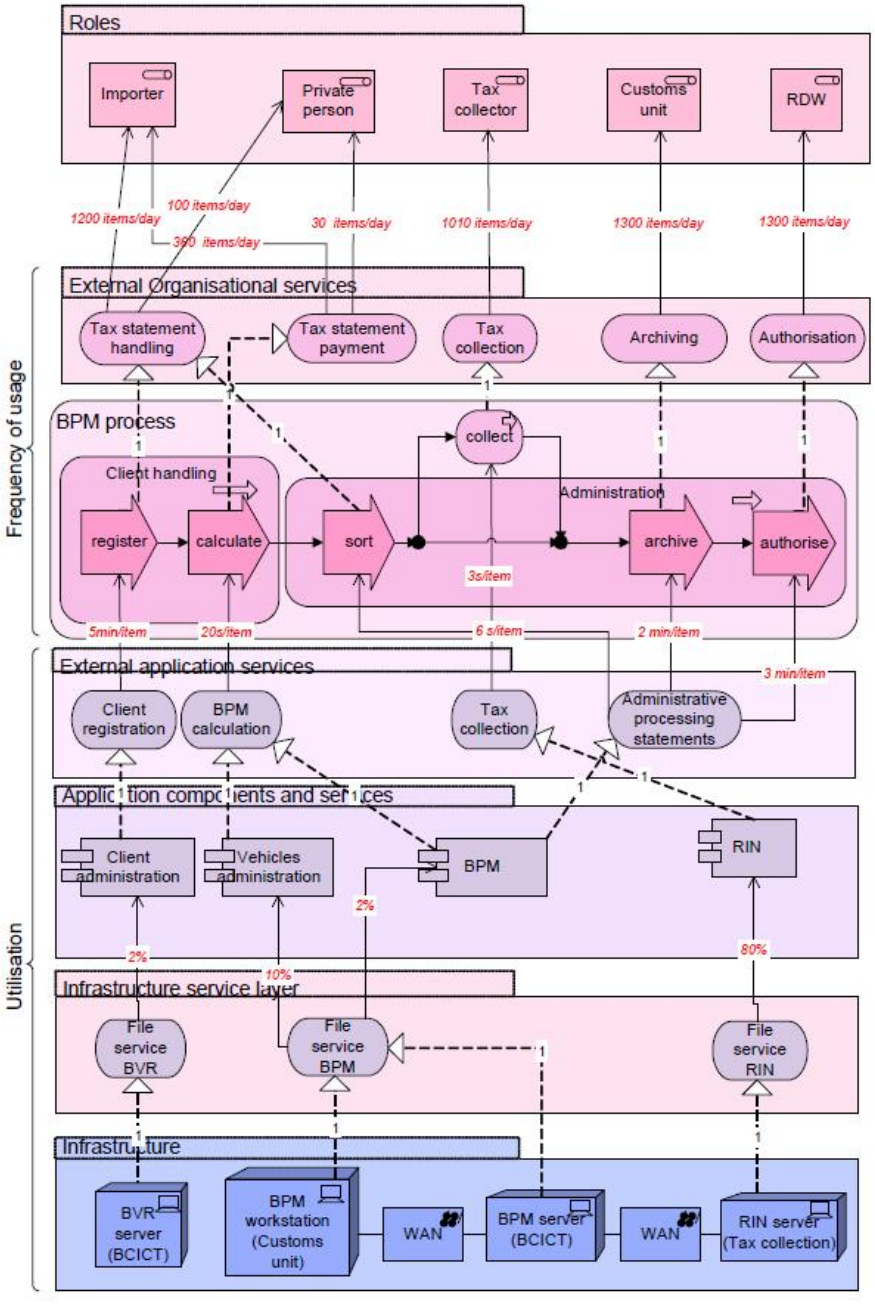

Fig. 3. ArchiMate model of the case Car Import

3 DEMO

3.1 Summary of DEMO

DEMO relies fully on the Ψ-theory [4]. In this theory, an enterprise (organization) is a system in the category of social systems [2]. The distinctive property of social systems is that the active elements are human beings or subjects. These subjects perform two kinds of acts: *production* acts (P-acts for short) and *coordination* acts (C-acts for short). By performing P-acts the subjects contribute to bringing about the goods or services that are delivered to the environment. By performing C-acts subjects enter into and comply with commitments towards each other regarding the performance of P-acts. Examples of C-acts are "request", "promise" and "decline". The effect of performing a C-act is that both the performer and the addressee of the act get involved in commitments regarding the bringing about of the corresponding P-act.

C-acts and P-acts appear to occur as steps in a generic coordination pattern, called *transaction*. Fig. 4 exhibits the basic transaction pattern (upper right corner), as the elaboration and formalization of the workflow loop as proposed in [3], which is drawn in the upper left corner. A transaction evolves in three phases: the order phase (O-phase for short), the execution phase (E-phase for short), and the result phase (R-phase for short). In the order phase, the initiator and the executor negotiate for achieving consensus about the P-fact that the executor is going to bring about. The main C-acts in the O-phase are the request and the promise. In the execution phase, the P-fact is brought about by the executor. In the result phase, the initiator and the executor negotiate for

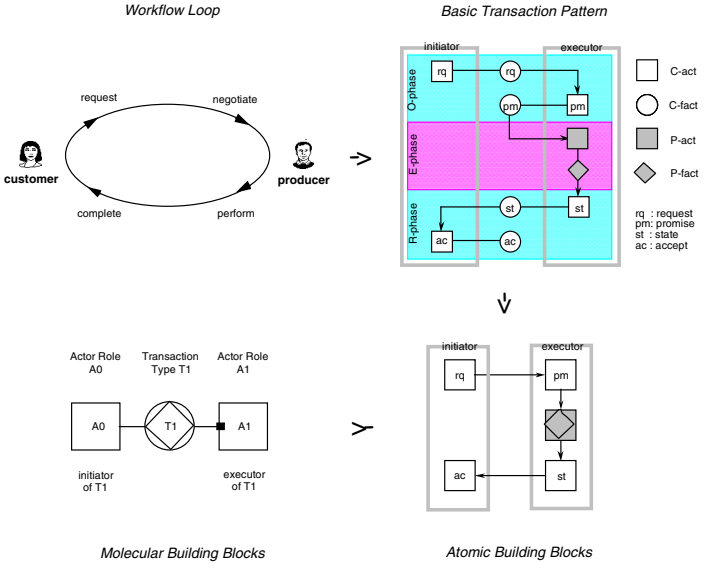

Fig. 4. Ontological building blocks of an organization

achieving consensus about the P-fact that is actually produced (which may differ from the requested one). The main C-acts in the R-phase are the state and the corresponding accept. The terms "initiator" and "executor" replace the more colloquial terms "customer" and "producer". Moreover, they refer to actor roles instead of subjects. An *actor role* is defined as the authority and responsibility to be the executor of a transaction type. Actor roles are fulfilled by subjects, such that an actor role may be fulfilled by several subjects and a subject may fulfill several actor roles.

The actual course of a transaction may be much more extensive than the basic pattern in Fig. 4. This is accommodated in the Ψ-theory by appropriate extensions of the basic pattern. At the lower right side of Fig. 4, a comprised notation is shown of the basic transaction pattern. A C-act and its resulting C-fact are represented by one, composite, symbol; the same holds for the P-act and the P-fact. At the lower left side the complete transaction pattern is represented by only one symbol, called the transaction symbol; it consists of a diamond (representing production) embedded in a disk (representing coordination). Transaction types and actor roles are the *molecular* building blocks of business processes and organizations, the transaction steps being the *atomic* building blocks.

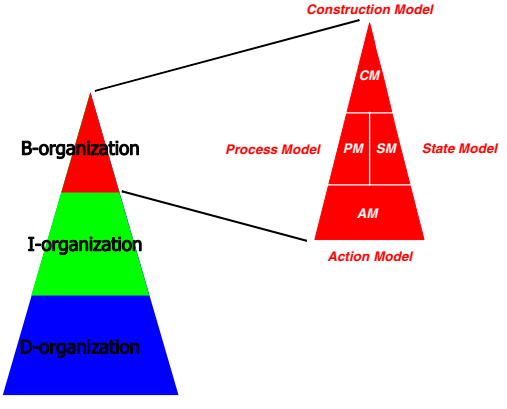

Fig. 5. The three aspect organizations

Another important component of the the Ψ-theory is the distinction between three human abilities, which are exerted both in C-acts and in P-acts: the forma, the informa, and the performa ability. Regarding coordination, the forma ability concerns uttering and perceiving written or spoken sentences, the informa ability concerns formulating thoughts and educing them from perceived sentences, and the performa ability concerns getting engaged in commitments. On the production side, the forma ability concerns datalogical production (storing, transmitting, copying etc. of data), the informa ability concerns infological production (computing, reasoning), and the performa ability concerns bringing about original new facts (deciding, judging, creating); we therefore call it ontological production.

The distinction between the three human capabilities on the production side gives rise to the distinction of three layered aspect organizations, as depicted in Fig. 5. By definition, the ontological model of an enterprise is the model (according to the Ψ-theory) of

its B-organization. DEMO helps in 'discovering' an enterprise's ontological model, basically by re-engineering from its implementation, as e.g. contained in a narrative description. The complete ontological model of an enterprise consists of four aspect models (see Fig. 5). The Construction Model contains the actor roles and transaction kinds, the Process Model contains the business processes and business events, the State Model contains the business objects and business facts, and the Action Model contains the business rules.

3.2 Analysis of the Case Car Import with DEMO

Every experienced DEMO analyst has his or her own way of working in producing the DEMO models of a case, being fully guided by the Ψ-theory. For novice DEMO analysts, however, a six-step method has been developed [4]. Applying the first steps of this method to the narrative description of the case Car Import produces the result as presented hereafter.

In The Netherlands any [imported car] is subjected to special kind of taxation called BPM. The business architecture supporting the whole collection process and the interaction of the [Dutch Tax Department (Belastingdienst)] with the importers and a number of other parties is described below. The [importer] (private person or car dealer/importer) must announce himself at the customer counter in any of the 30 Customs units in The Netherlands with the <imported vehicle>, its (<provenance>) documents, the <approval proof of its technical inspection>, and possibly with cash for the payment of the BPM tax. The public servant will handle the tax declaration as follows: first he will check all the documents, then he will fill in all the data into a client BPM application (running on a local server) and will calculate the due BPM tax value (using the BPM application and the catalogue value for that particular car). One copy of the BPM form (BPM17 ex 1) will be issued and sent to the administration. Another copy of this form is handed to the importer (BPM17 ex3), together with either the (evidence of a cash payment) (if the importer is able to pay the BPM amount in cash), or with (a bill ("acceptgiro")) issued for the due amount (in the case the importer is not able to pay in cash).

At each Customs unit there will be [public servants] assigned to handle the additional administrative operations regarding all the incoming (BPM statements). Once a day, this person will collect all the incoming BPM17 forms. For ones, which were <paid> in cash, he will issue and <authorize> another copy of the BPM form (BPM17 ex2). This copy will be sent to RDW ("Rijksdienst voor het Wegverkeer" - the Netherlands Road Transport Department), which keeps the evidence of <all registered vehicles> in The Netherlands. The first copy of BPM 17 will be then sent to the archive. The forms which are not yet paid, are kept "on hold" until they are paid. The payment administration and the notification service for (all incoming payments) for these BPM forms is done by a separate department of the Belastingdienst, namely the Tax Collection Departments ("[Inning]"), which is responsible for the collection of all <payments> via bank. Once such (a notification is received) (via the BPM server application) the administration will prepare, <authorize> and send the copy of BPM17 ex.2 to [RDW], and will permanently archive the ex1 of the BPM17.

All ontological things are underlined. In addition, actors are indicated by placing their name between "[" and "]", P-acts and –facts are indicated by placing their name between "<" and ">", and C-acts and –facts are indicated by placing their name between "(" and ")". Next, we put the transaction pattern 'over' the onological things. This results in the identification of three transaction kinds: T01 - the import of a car, T03 - the admission of a car to the Dutch road network, and T04 - the payment of the BPM tax.

T01 is actually outside the scope of the case, but we will start to include it in our model since it clarifies the whole process from importing a car through to admitting it to the road network, and since paying BPM tax will turn out to be disconnected from importing a car, although the case description suggests so. T03 is only slightly mentioned, namely in the last sentence: ... *the administration will prepare, authorize and send the copy of BPM17 ex.2 to RDW* ... This sentence, inparticular the term "authorize" suggests that the sending of the copy counts as requesting for admission to the road network. However, this cannot be the case from an ontological point of view: only a car owner is authorized to request for admitting the car to the road network, as also only a car owner is authorized to request for importing the car.

Another, related, sentence that is ontologically puzzling, is the third one: *The importer (private person or dealer/importer) must announce himself at the customer counter in any of the Customs units* ... The question to be answered is "Who is requesting the importer to pay the BPM tax?". A candidate actor role is the one that decides on the import of a car. However, although the case description suggests that paying the BPM tax is connected to importing a car, this is not true, as further investigation has learnt us. The tax one has to pay as a prerequisite for importing a car is the VAT. We have included this transaction for completeness sake (T02). Importing a car however is distinct from getting it admitted to the road network! One could do the first and omit the second. So, there must be another actor role that requests to pay the BPM tax. Since paying this tax is a prerequisite for getting the car admitted to the road network, it is obvious (and institutionally quite correct) that RDW requests the car owner to pay the BPM tax after the car owner has requested the RDW to admit the car to the road network. Concludingly, we arrive at the Actor Transaction Diagram as exhibited in figure D1. The corresponding Transaction Result Table is shown in Table D1. Together they constitute the Construction Model (Fig. 5).

As said before, the left part of Fig. 6 was only included for the sake of explaining clearly the distinction between importing a car (including paying the VAT) and admitting a car to the road network (including paying the BPM tax). Fig. 6 clearly shows that the two processes are disconnected. Therefore, we only produce the Process Model for the right part, T03 and T04 (see Fig. 7). As a help in understanding it we have added to each step the person or organizational unit or institution that actually performs the step. For the sake of simplicity we have chosen CA03 and CA04 to be fulfilled by a private person. Obviously, RDW is the authorized institution for fulfilling actor role A02 (road network admitter). However, for performing T04/ac it has apparently delegated its authority to the Tax Office (Belastingdienst). The dashed arrow from T04/ac to T03/ex means that RDW has to wait for deciding to admit a car to the road network until the BPM tax has been paid. From the case description we derive that the current way in which RDW is informed about this fact by the Belastingdienst is the sending of the copy of BPM17 ex.2.

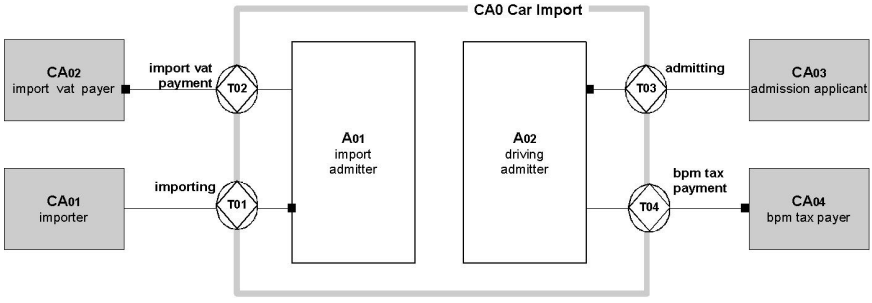

Fig. 6. Actor Transaction Diagram (ATD) of the case Car Import

Table 1. Transaction Result Table of the case Car Import

Transaction type	Transaction result
T01 importing	R01 *Import I has been performed*
T02 import vat payment	R02 *import vat for Import I has been paid*
T03 admitting	R03 *Admission A has been started*
T04 bpm tax payment	R04 *BPM Tax for admission A has been paid*

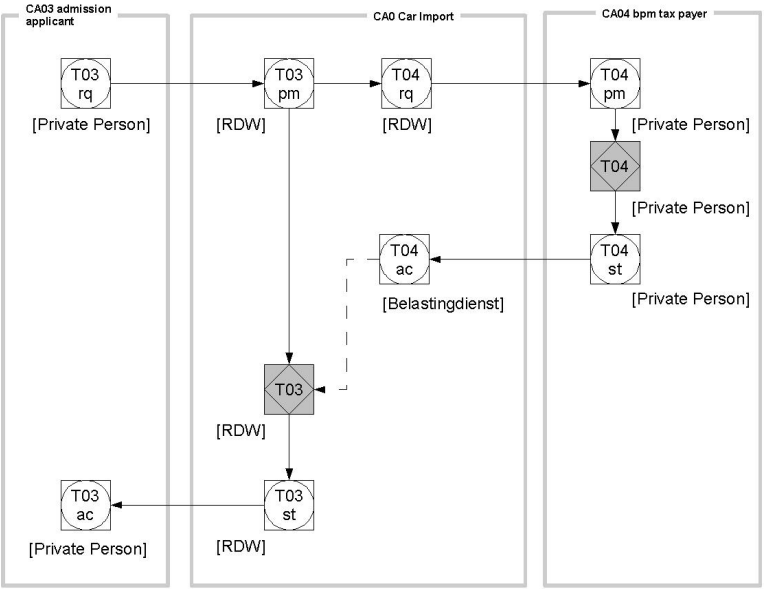

Fig. 7. Process Model (PM) of the case Car Import

4 Comparing ArchiMate and DEMO

4.1 Theoretical Comparison

By the theoretical comparison of ArchiMate and DEMO we mean the comparative evaluation of the Way of Thinking as well as the Way of Modeling of each, in accordance with the evaluation framework that is known as the 5-way model [13].

By the Way of Thinking of an approach is understood its theoretical foundation, in particular the basic understanding of the object of analysis, in our case the enterprise. At first sight, the business layer in ArchiMate seems to correspond with the B-organization in DEMO. This appears not to be true, however. To clarify the difference, we present in Fig. 8 the relationship between an organization and its supporting ICT-systems as conceived in DEMO.

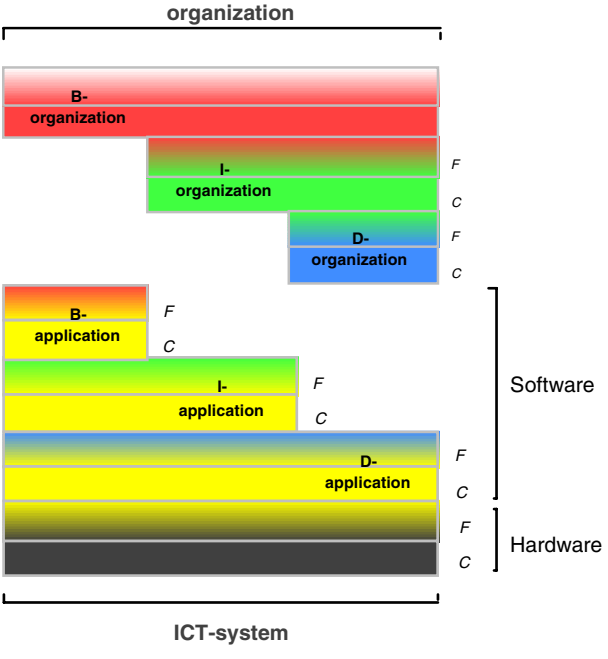

Fig. 8. Organization and ICT-system

Apparently, the business layer in ArchiMate corresponds to the three organization layers in DEMO (B-, I- and D-) collectively. Most probably, the application layer in ArchiMate corresponds to the B-application and the I-application layer in DEMO, and the technology layer in ArchiMate corresponds to the D-application and the hardware layer in DEMO. However, since we have focused on the business layer, there is no evidence to verify or falsify this hypothesis. Next, the Ψ-theory underlying DEMO provides for an appropriate and rigorous foundation. ArchiMate lacks such a foundation.

As a consequence, the semantics of the meta model (cf. Fig. 1) is undefined, which may easily lead to misinterpretations.

By the Way of Modeling (WoM) of an approach, is understood the definition of the distinct models of an enterprise, their representations, and the derivation of the models from a case description. The WoM of ArchiMate is to look for terms in the case description that designate instances of the meta model concepts. In this manner, the model as represented in Fig. 3 is produced. This WoM resembles very much the one in several modeling techniques from the seventies, of which the ER model is probably the best known. The advocated WoM was to look for nouns and verbs in a text. Nouns were taken as names of entity types and verbs ware taken as names of relationship types.

Despite the fact that ArchiMate's WoM (and meta model) is widely used, it has serious drawbacks. One is that irrelevant concepts are included in the model, just because there was a term in the description referring to it. Another one is that relevant concepts are missing because references in the description were forgotten. A third one is that different analysts produce different models, since the meta model is multi-interpretable.

In contrast, since the four aspect models of DEMO (cf. Fig. 5) are grounded in the Ψ-theory, the ontological model of an enterprise is guaranteed coherent, consistent, comprehensive, and concise. Moreover, it only shows the essence of the enterprise, completely independent of any implementation issue. This not only holds for the B-organization (the essence of the enterprise) but also for the I-organization and the D-organization. Therefore, it the ideal starting point for the re- design and re- engineering of enterprises [5]. Lastly, different analysts will produce the same ontological model, because every enterprise has only one such model.

4.2 Comparing the Analysis Results

As one will have observed, the results of applying ArchiMate and DEMO to the case Car Import, as presented and discussed in Section 2 and Section 3, differ very much. This is due to the differences between the Way of Thinking and the Way of Modeling of the two approaches. Obviously, ArchiMate takes the case description literally; it is its Way of Modeling. On the other hand, the DEMO analysis has evoked 'critical' questions. They have lead to a profound understanding of what is essentially going on in the described situation.

First, importing a car and getting a car admitted to the Dutch road network are distinct and disconnected processes. Only for the latter one, there is the prerequisite that the BPM tax is paid.

Second, it is not true that the Tax Office authorizes the RDW to admit a car to the road network; instead it is the car owner who requests for admission. The Tax Office only has a delegated authority in accepting the results of transactions T04 (BPM tax payment). Subsequently it informs the RDW that the payment has been received. We have a strong suspicion that the Tax Office and the RDW are not aware of these essential relationships. It is for sure, however, that this ignorance causes a lot of confusion and many failures in attempts to make the processes more efficient, e.g. by applying modern ICT.

5 Conclusions

We have carried out a comparative evaluation of ArchiMate and DEMO, both theoretically and practically, i.e. on the basis of the analysis of the same case by each approach. Space limitations prohibit us from giving a full and detailed account of our research. Only the most noticeable issues could be presented and discussed. In addition, a thorough assessment of the strengths and weaknesses of ArchiMate and DEMO can only be performed on the basis of multiple real-life and real-size cases, taken from different areas. Nevertheless, some conclusions can certainly be justified already now.

The first conclusion is that ArchiMate and DEMO are hardly comparable, for several reasons. One is that ArchiMate is a second wave approach, whereas DEMO is a third wave approach, as was discussed in Section 1 already. Another reason is that DEMO is founded in a rigorous and appropriate theory, whereas ArchiMate lacks such a foundation. Therefore, its semantics are basically undefined, which unavoidably leads to miscommunication among Archimate users. One would expect that having a rigorous semantic definition would be a prerequisite for an open standard.

A second conclusion regards the abstraction layers as disinguished by ArchiMate and DEMO. DEMO (in fact the Ψ-theory) makes a well defined distinction between three abstraction layers: the B-organization, the I-organization, and the D-organization. Only in the B-organization original new production facts are brought about (deciding, judging, manufacturing etc.), by which the enterprise world is changed. In the I-organization one computes, calculates, reasons; this does change the world. In the D-organization one stores, copies, transports etc., documents. Despite the fact that ArchiMate belongs to the second wave, it does not make a distinction between infological and datalogical issues in the business layer. As an illustration of the point, the model in Fig. 3 includes actions like archiving and sorting, next to calculation. Although this seems not to be an issue of worry for Archimate, we think ArchiMate could profit from solidly incorporating this distinction. It would make Archimate to some extent suitable for re-engineering projects. The lack of a rigorous semantic definition remains a major obstacle for actually doing it.

Although ArchiMate and DEMO are to a large extent incomparable, we think that they can usefully be combined. As a matter of fact, several studies have been carried out concerning the combination of DEMO with some second generation approach, since DEMO does not really cover the implementation of an organization. An interesting study in this respect is an evaluative comparison of DEMO and ARIS, in particular the EPC (Event Process Chain) technique [14]. As one of the practical outcomes, a procedure has been developed for producing EPCs on the basis of DEMO models. In this way, the rigorous semantics of DEMO are so to speak enforced upon the EPC. We conjecture that such a combination is also possible and beneficial for ArchiMate and DEMO.

References

1. Austin, J.L.: How to do things with words. Harvard University Press, Cambridge (1962)
2. Bunge, M.A.: Treatise on Basic Philosophy. A World of Systems, vol. 4. D. Reidel Publishing Company, Dordrecht (1979)

3. Denning, P., Medina-Mora, R.: Completing the loops. In: ORSA/TIMS Interfaces, vol. 25, May 3-June, pp. 42–57 (1995)
4. Dietz, J.L.G.: Enterprise Ontology – theory and methodology. Springer, Heidelberg (2006)
5. Dietz, J.L.G.: Architecture – building strategy into design, Sdu Netherlands (2008)
6. Special issue of Communications of the ACM 49(5), 59–64 (May 2006)
7. Habermas, J.: Theorie des Kommunikatives Handelns, Erster Band. Suhrkamp Verlag, Frankfurt am Main (1981)
8. Hoogervorst, J.A.P., Dietz, J.L.G.: Enterprise Architecture in Enterprise Engineering. Enterprise Modelling and Information Systems Architecture 3(1) (March 2008)
9. Iacob, M.-E., Jonkers, H.: Quantitative Analysis of Enterprise Architectures. Enschede: Telematica Instituut, Archimate Deliverable 3.5.1b/v2.0. TI/RS/2004/006 (2004)
10. Langefors, B.: Information System Theory. Information Systems 2, 207–219 (1977)
11. Lankhorst, M., et al.: Enterprise Architecture at Work. Springer, Heidelberg (2005)
12. Searle, J.R.: Speech Acts, an Essay in the Philosophy of Language. Cambridge University Press, Cambridge (1969)
13. Seligman, P.S., Wijers, G.M., Sol, H.G.: Analyzing the structure of I.S. methodologies; an alternative approach. In: Maes, R. (ed.) Proceedings of the First Dutch Conference on Information Systems, Amersfoort (1989)
14. Strijdhaftig, D.: DEMO and ARIS – developing a consistent coupling, Master Thesis TU Delft (2008)
15. Weinberg, G.M.: An Introduction to General Systems Thinking. John Wiley & Sons, Chichester (1975)
16. Wittgenstein, L.: Tractatus logico-philosophicus. Routledge & Kegan Paul Ltd., London (1922) (German text with an English translation by C.K. Ogden)