Integrating Value Modelling 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. Present day enterprises often become service-oriented enterprises, which are comprised of a dynamic network of organisations that collectively provide services. These services express stakeholders' needs, and can be viewed from different perspectives. In this paper, we consider value web and enterprise architecture perspectives. Specifically, we present a step-wise mapping approach that integrates the value modelling technique e^{3}value into the enterprise architecture language ArchiMate. The main contribution of this paper is twofold. First, we present our initial arrays into how to bridge between *e^{3}value* and ArchiMate and, in doing so, we show how these modelling techniques complement one another. Second, by reflecting on the bridging between *e^{3}value* and ArchiMate, we discuss the limitations of our integration, which provides useful input for future efforts into model integration. We illustrate our approach by means of a case in the insurance domain.

Key words: enterprise architecture, value web, ArchiMate, e^{3}value, model integration

1 Introduction

Enterprises need to negotiate many challenges, such as changes in the economic climate, mergers, acquisitions, novel technologies, et cetera. As a result, enterprises need to be agile to improve their chances of survival. Moreover, the shift towards a services oriented economy makes it even more important for enterprises to adopt themselves to changes. Present day enterprises often become service-oriented enterprises, which are comprised of a dynamic network of organisations that collectively provide services [1].

Dealing with such changes requires a good steering instrument. Such a steering instrument should provide enterprises with the ability to analyse the current state of the enterprise, identify and describe alternative future states, guard the cohesion and alignment between the different aspects of an enterprise such as business processes and their ICT (Information and communications technology) support. Enterprise architecture is generally considered to provide such a mechanism for cohesive steering [2, 3, 4]. It aims to provide management with appropriate indicators and controls to steer the transformation of an enterprise into the desired direction. As such, enterprise architecture is concerned with the enterprise as a whole and not just enterprise-wide ICT architecture.

Although the value of architecture has been recognized by many organisations, mostly separate architectures are constructed for various organisational domains which remain unspecified or implicit [5]. In general, business models focus on the service value generated by a business, whereas enterprise architecture models show how a business realizes these services [5]. Ensuring integration between the conceptual models representing the different stakeholder perspectives is a research problem. First and foremost, such a model integration is needed to *ensure consistency*. Different modelling techniques emphasize different perspectives on the *same system*, and so should be in line with one another [6]. As such, model integration fosters *traceability*. For example, considering parts of an organisation modelled in one perspective, such as from a value perspective the actors exchanging value, it may return in another perspective, such as, from an ICT perspective, the actors exchanging messages.

In this paper we present our first arrays into integrating, on a syntactic level, two conceptual models: e^3*value*, for modelling a value perspective on the business at hand, and ArchiMate, for modelling business operationalization (such as business processes and the underlying ICT realizing the business collaboration). It makes sense to look specifically at integrating e^3*value* into ArchiMate: ArchiMate has recently been adopted by the Open Group [1] as a standard for modelling enterprise architectures, but, as we shall see in this paper, lacks expressivity for modelling an enterprise from a value perspective.

To the best of our knowledge, only [5] have so far considered specifically the integration of e^3*value* and ArchiMate. However, while [5] do argue that ArchiMate complements e^3*value* in terms of profitability calculations, little effort is made to show exactly *how* to bridge the two modelling techniques, and what the limitations in such a mapping are. More specifically, while [5, p.8] do provide a mapping between the e^3*value* and ArchiMate metamodels, they leave implicit the actual use of this mapping to translate an e^3*value* model into ArchiMate and vice versa. Worse still, [5] fail to discuss the conceptual differences between the e^3*value* and ArchiMate metamodels, thus leaving for example implicit that e^3*value* and ArchiMate interpret the concept of an "Actor" differently. However, ArchiMate concepts sometimes have no place in a e^3*value* model and vice versa. For example: an IT-department that is not profit-and-loss responsible may be modelled as an Actor in ArchiMate but, because it is not interesting from a value perspective, should not be present in an e^3*value* model.

Note that in this paper, we present an *initial* mapping of e^3*value* and ArchiMate on a *syntactical level*. As such, the contribution of this work is twofold: (1) to provide an initial mapping between the value modelling technique e^3*value* and the enterprise modelling technique ArchiMate. In this mapping, we show *how* e^3*value* and ArchiMate complement one another. and (2) following up on that initial mapping, to point out differences between the two modelling techniques from both syntactic and semantic levels, which provides useful input for

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

further model integration. We use a running example of an insurance scenario to illustrate our ideas.

The remainder of this paper is organized as follows. Section 2 introduces the insurance case study. In section 3, we discuss the integration of ArchiMate and e^3*value* models. In particular, we show how ArchiMate operationalizes a value web modelled in e^3*value*, and how e^3*value* can be used for profitability calculations of an operationalization modelled in ArchiMate. Section 4 points out the initial mapping of our approach and discuss functionalities and limits of such integration. Section 5 presents related work. Section 6 concludes.

2 The Archinsurance Value Web

For illustration purposes, we present a fictitious but realistic insurance case. This case is inspired by a paper on the economic functions of insurance intermediaries [7], as well the running insurance case that illustrates the ArchiMate language specification [8, 9]. In this section, we first use our insurance case to introduce the value modelling technique e^3*value* (Sect. 2.1). Subsequently, in Sect. 2.2, we discuss the particular insurance case scenario that will be used as a running case for the remainder of this paper.

2.1 Modelling the Archinsurance value web in e^3*value*

Archinsurance, a large insurance company, offers one of its many products, car insurance, directly to its customers via the internet and its sales representatives. The reason for selling directly to customers is mainly to cut costs that come with an intermediary, such as premiums paid and additional administrative costs. These cost cuttings, in turn, can be passed on to the customer in terms of a lower insurance fee.

We use e^3*value* to model this direct-to-customer sales model. e^3*value* focuses on modelling the value exchanges between actors participating in a value web, depicting what each actor offers to others, and what it receives in return. The principle of economic reciprocity, which states that an offering from an actor should always be compensated, is central to e^3*value*: in other words, one good turn deserves another.

This Archinsurance direct-to-customer business model is depicted in the e^3*value* model in Fig. 1. Here, we see that:

- A customer (modelled as a market segment) provides his ‘complete risk profile’ to Archinsurance, which is valuable to the Archinsurance (an actor) since it defines the basis for the offered insurance package and its associated fees.
- Archinsurance transforms the risk profile in an insurance offering via ‘Contracting’ (a value activity), and the customer receives a car insurance from Archinsurance in return.

Observe here the economic reciprocity, as modelled by a value interface in e^3*value*: Archinsurance only provides its car insurance if it receives in return another valuable object: a complete risk profile.

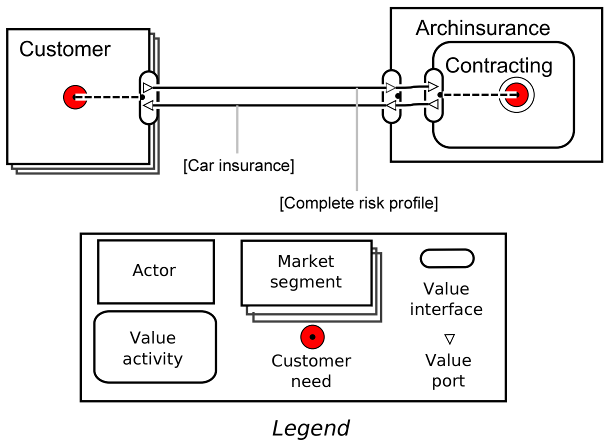

Fig. 1. Value model of direct sales of car insurance

2.2 Archinsurance: moving from a direct-to-customer to an intermediary sales model

Archinsurance finds that its direct-to-customer model brings with it also the problem of *adverse selection of risk profiles* (cf. [7]). This adverse selection means that Archinsurance offers customers a car insurance against an inappropriate fee, or worse still, offers a customer a package that may in fact not be issued, because of an incomplete or faulty risk profile on the part of the customer. This is mainly due to a problem of information asymmetry: the customer knows more about his risk profile than the insurer does. For instance, customers knows more about their past accidents, or medical history. However, due to a lack of expertise or on purpose, customers do not always provide the necessary risk profile to the insurance provider.

To reduce the risk of receiving incomplete customer profiles, Archinsurance considers moving towards selling insurances via an *intermediary* (see Fig. 4). The rationale for introducing an intermediary, and how this can be beneficial for all parties involved, are explained as follows:

– The customer provides 'personal information' to the intermediary, which is valuable to the latter because it allows for the composition of a complete risk profile.

- Using the value activity ‘Create customized insurance package’, the intermediary matches customer information to appropriate offerings, and possibly requests additional customer information. Note that the value model shows only the (high level) value-adding activities: it can be detailed in an operational ArchiMate model afterwards (see Sect. 3.2).
- The intermediary provides Archinsurance with a ‘Complete risk profile’, which, as discussed, the insurer may use to mitigate adverse selection of risk profiles.
- In return for the profile, Archinsurance pays the intermediary a premium.
- The customer receives a ‘Tailored car insurance guarantee’ from the intermediary, and the actual ‘Car insurance’ from Archinsurance in return. This is to depict the advantage that, through intervention of the intermediary, the customer receives an insurance package in line with his profile, against an appropriate fee.

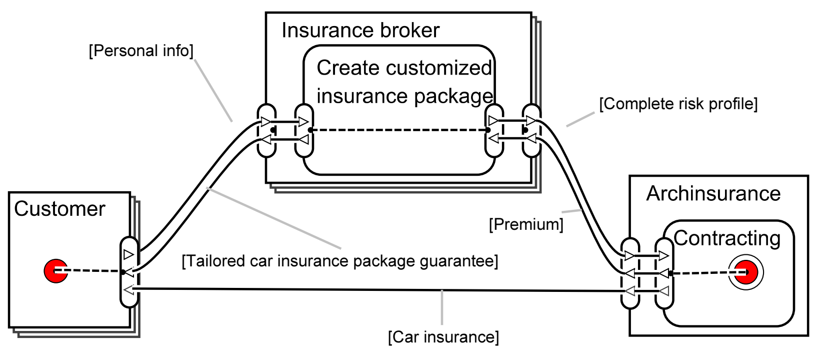

Fig. 2. Value model of car insurance sales via an intermediary

3 Towards Integrating ArchiMate and e^3*value*

We now show our approach for mapping between ArchiMate and e^3*value*, using the baseline e^3*value* models defined thus far as input.

We first briefly introduce the enterprise architecture modelling technique ArchiMate (Sect. 3.1). Thereafter, we discuss how to map a e^3*value* model to an ArchiMate model (Sect. 3.2) and, vice versa, how to map an ArchiMate model to e^3*value*(Sect. 3.3).

3.1 Modelling an enterprise architecture in ArchiMate

We rely on the ArchiMate modelling language to model the enterprise architecture of the Archinsurance case. ArchiMate has been transferred to the

Open Group, where it is slated to become the standard for architectural description accompanying the Open Group's architecture framework TOGAF [4]. ArchiMate focuses on administrative sector, unlike UPDM (Unified Profile for DoDAF/MODAF) whose main focus is the defense sector [10]. ArchiMate is geared towards "Information processing dominant organisations" such as banks, insurance companies, government agencies, et cetera [11]. In this paper, it offers a coherent description of the enterprise architecture to enable communication among stakeholders, and to guide change processes within Archinsurance.

3.2 e^3*value* to ArchiMate: operationalization of a business scenario

For the direct-to-customer value web in Fig. 1, we provide an enterprise architecture in Fig. 3. From e^3*value* we arrive at this enterprise architecture modelled in ArchiMate in two steps:

1. Concept inheritance from e^3*value* to ArchiMate: We find that actors and, in particular, business functions, provide a bridge between e^3*value* models and ArchiMate models. *Actors and market segments* in e^3*value* become *actors* in ArchiMate. In the Archinsurance case, the actors 'Customer' and 'Archinsurance' from the e^3*value* model (Fig. 1) become actors in the ArchiMate model (Fig. 3). *Value activities* in e^3*value* become *business functions* in ArchiMate. In the Archinsurance case, the value activity 'Contracting' becomes a business function in the ArchiMate model. Finally, given a e^3*value* model wherein actors execute value activities, we can relate business functions to actors in ArchiMate. In the Archinsurance case, we see that Archinsurance executes the value activity 'Contracting' and so can link in ArchiMate 'Contracting' to the relevant business actor (i.e., Archinsurance).

2. Operationalization of business functions in ArchiMate: We find that business functions, which in ArchiMate denote the high-level functions that a company executes, provide an excellent starting point for modelling operational details. For example, we detail exactly what *operational* business process steps realize the business function 'Contracting': the steps 'Registration of customer profile', 'Eligibility check' and 'Estimate monthly customer fee'. Subsequently, we can use these business process steps to model the required ICT support in ArchiMate. For example, the enterprise architecture that realizes the direct-to-customer scenario, we see that the IT service 'Customer administration service' is required for the registration of a customer profile, and that a 'Risk assessment service' is required for the eligibility check and the estimation of the monthly customer fee (see Fig. 3). Finally, note that ArchiMate allows for also modelling the physical ICT infrastructure required to run said ICT-applications. Due to space restrictions however, we do not show this physical infrastructure.

In ArchiMate, a layered view provides a natural way to look at service-oriented models. Services define externalized functionality from one layer that is useful in another layer. This shows the service-orientation of ArchiMate: no

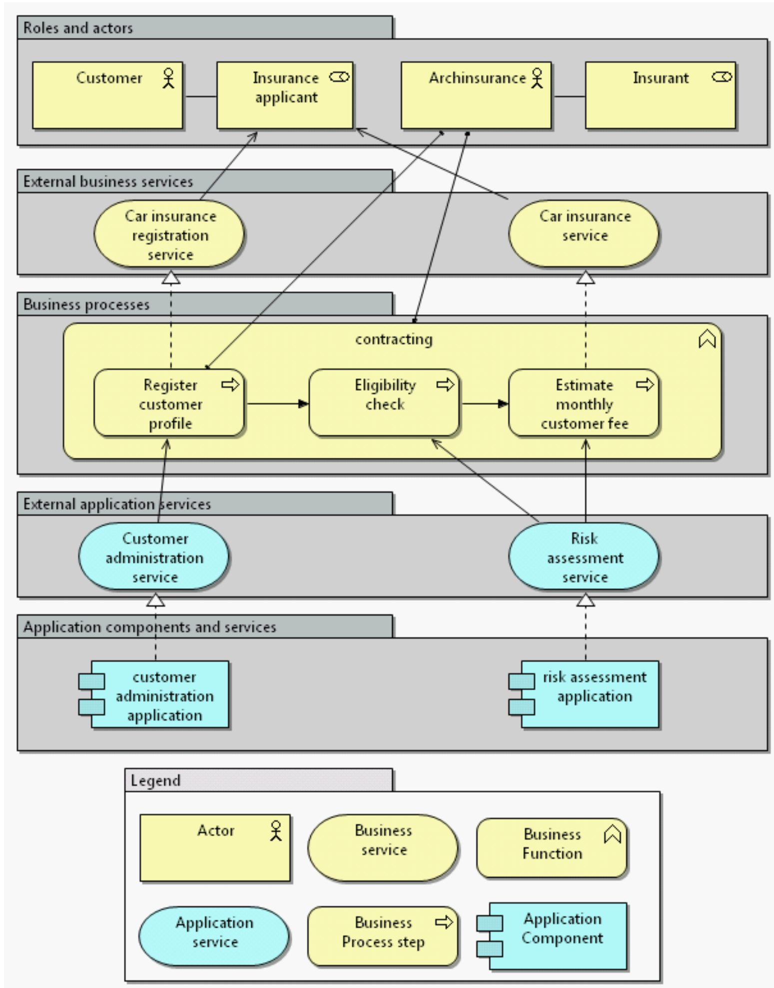

Fig. 3. (Partial) enterprise architecture model: Direct sales of insurance

matter what changes in a layer (e.g., business or application), the services offered from one layer to another remain the same.

Note that the details discussed regarding the operational business process steps, applications and physical infrastructure are especially useful for cost estimation of an ICT infrastructure (see Sect. 3.3). Moreover, it is important to note here is that value objects, which e^3*value* uses to show the economic rationale for actors to participate in a value web, are not carried over to ArchiMate. We return to this point in our reflection on mapping e^3*value* to ArchiMate (Section 4).

3.3 ArchiMate to e^3*value*: profitability of operationalization

Consider now the ArchiMate model in Fig. 4, which depicts an operationalization in terms of needed business processes and ICT infrastructure for the introduction of an intermediary. From this model, we can derive an e^3*value* model to calculate the profitability of the operationalization modelled in ArchiMate. We do this in three steps:

1. Annotate expenses in ArchiMate: We can annotate the ArchiMate model that includes an intermediary (Fig. 4) with the required expenses. Examples include expenses for (1) the IT application 'Customer registration' of the intermediary. The intermediary requires this application to enable interoperability with Archinsurance. And (2) the business process step 'Customer profile registration' of the intermediary. The intermediary requires training for this business process step, in particular to know what type of customer data Archinsurance demands for its car insurance. Note that we do not show the actual expenses in the enterprise architecture model because of the fictitious nature of this case.

2. Import expenses in e^3*value*: As in Sect. 3.2, we use business functions and roles/actors from ArchiMate to make a bridge towards value activities and actors in e^3*value*. For the enterprise architecture of the Archinsurance case with an intermediary in Fig. 4, we arrive at the value web modelled in e^3*value* in Fig. 2 as follows:
(1) *Aggregating ICT-infrastructure expenses and business process expenses onto a business function level.* We can aggregate expenses into a business function because ArchiMate shows exactly what dependencies exist between different layers. For example, the enterprise architecture model in Fig. 4 shows that the business function 'Create customized insurance package' relies upon the business process step 'Customer profile registration', to which - as mentioned - training expenses are attached. In turn, the business process step 'Customer profile registration' is realized by the application 'Customer administration' (via the service customer administration service intermediary), to which installation expenses are attached.
(2) *Translating the business functions and actors/roles in ArchiMate to* e^3value. After having aggregated the expense carriers into business functions in ArchiMate, we can map the enterprise architecture model to e^3*value*(while keeping the costs intact of course for the profitability calculations). For the Archinsurance case, the actors 'Customer', 'Insurance broker' and 'Archinsurance' become actors in e^3*value*. Thereafter, the actor 'Intermediary' receives the value activity 'Create customized insurance package' (from its similarly named business function in ArchiMate), while the actor 'Archinsurance' receives the value activity 'Contracting'. For the intermediary value web, see an example in Fig. 5. We observe that ArchiMate can relate business functions to actors (as for example shown by line between Archinsurance and the business function Contracting). Therefore, we know what actor performs what business function and can import

this knowledge into e^3*value*.

3. Perform profitability calculations in e^3*value*: With e^3*value*, we can now perform profitability calculations. e^3*value* allows for this by (1) Introducing time series that depict the evolution of a value web over different points in time. Besides the initial Archinsurance value web in Fig. 2, we may model the monthly recurrence of an insurance fee paid by the customer, and (2) Experimenting with variables, and observing the results over a time series. For example, one can calculate profitability from: (1) The initial investments in the initial e^3*value* model, and (2) The model depicting monthly payments by the customer. Hereby, one can experiment with the number of customers, as well as the time period, and see how this will affect the profitability of the service operationalization.

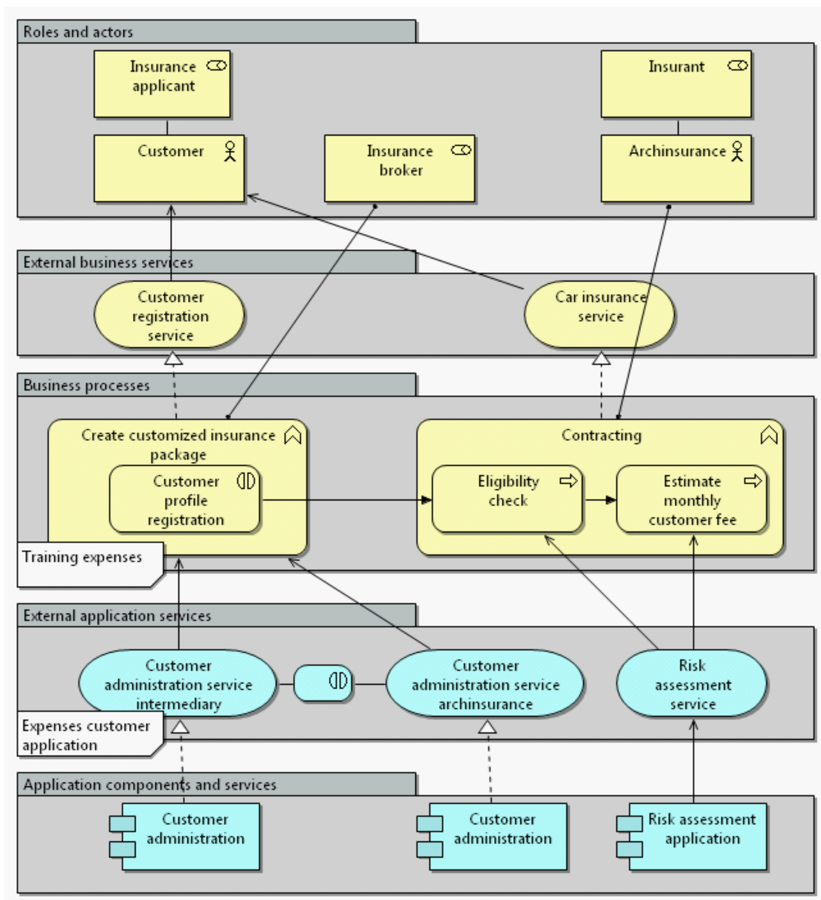

Fig. 4. (Partial) enterprise architecture model including an intermediary

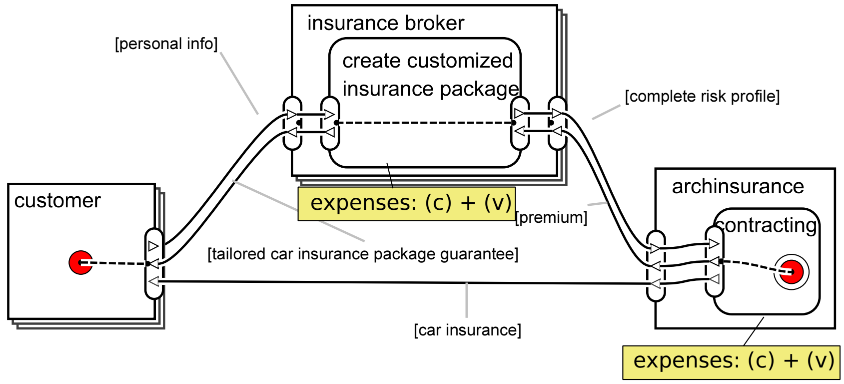

Fig. 5. Expenses in e^3*value*, based upon operationalization modelled in ArchiMate (c = fixed expense, v = variable expense)

4 Comparison of e^3*value* and ArchiMate Concepts

The mapping discussed so far presents a first step towards integrating e^3*value* and ArchiMate, as well as an illustration of why such model integration is useful. However, as stated in the introduction, this mapping represents only our initial attempt at model integration. As a result the mapping is naive, by for example assuming that the concept of an actor as defined in e^3*value* maps one-to-one to the concept of an actor in ArchiMate.

To address this naive concept mapping, and thus create input for future efforts towards integrating e^3*value* and ArchiMate, we now discuss conceptual differences between the aforementioned models. We base our discussion on lessons learned from the running ArchiMate case.

Actor concept. Consider the concepts of actor and market segment in e^3*value*, and actors and roles in ArchiMate. In e^3*value*, the idea of a "profit-and-loss responsible unit" is central to the concepts of actor/market segment (where a market segment can be considered as a "stack" of n actors, with n denoting the market segment's size). In other words, if an object cannot assume financial responsibility on its own, it would not be modelled in e^3*value*. ArchiMate, differently, considers actors as "active entities…that perform behaviour such as business processes or functions" [8, p. 5]. Thus, the notion of actors in ArchiMate is broader than that of e^3*value*, where the financial (profit-and-loss responsibility) criterion is not present, but only the criterion that the actor has some function to execute.

To illustrate, consider in the insurance scenario, 'Contracting' from the direct-to-customer Archinsurance model (see Fig. 4). In e^3*value*, it is shown that Archinsurance performs 'Contracting', because Archinsurance can assume its own financial responsibility. However, it may well be that *inside Archinsurance* contracting is performed by different departments. For example, checking the eligibility of an insurance application - as part of contracting - may very well be performed by a separate back office specialized in car insurances. For Archinsurance, the modelling of these different departments may actually be important since they may have an impact on how the ICT support is modelled. However, when converting back to e^3*value*, the modelling of departments as actors in ArchiMate would be problematic since they are not profit-and-loss responsible.

Service concept. The criterion of what should be considered as a service differs between ArchiMate and e^3*value*. Following the ArchiMate model for the Archinsurance case, business services are - for the customer - the point of contact with the enterprise, such as 'Car insurance registration service' and 'Car insurance service' (see Fig. 3). More specifically, cf. [8], a service is a unit of functionality that some entity makes available to its environment, and which also has value for some entities in its environment. e^3*value*, differently, considers services on a higher level of granularity. For example, only the fact that the customer receives car insurance would matter, not that there is a contact point service for registration of the car insurance service. The latter happens more on a transactional level.

Seeing that business services are an important concept in both frameworks (in ArchiMate for connecting the different architectural layers, and in e^3*value* given its service orientation in recent work [12]), it seems to be useful to further analyse the concept of a service. Thus, in future research into model integration we should explicitly take into account different levels of granularity: services as a transaction, where one models elementary transactions such as customer registration, and services as business services, where one models commercial services as they exist in the marketplace.

Value object concept. The different foci of e^3*value* and ArchiMate are again reflected in the difference between an e^3*value* value object and an ArchiMate business object. As with our previous discussion, the main difference between these two concepts arises from the focus of e^3*value* on modelling an enterprise's value perspective, with e^3*value* maintaining the criterion that "a value object can be a good, service . . . or even experience. *The important point here is that a value object is of value to one or more actors*" [13]. ArchiMate, differently, considers business objects to represent the "important informational or conceptual concepts in which the business thinks about a domain"[9], which in ArchiMate models often translates to business objects that have a direct relevance to the underlying business processes and information systems, such as 'invoice' and 'insurance policy'.

On the one hand, this difference in interpretation needs to be accounted for when converting an e^3*value* model to ArchiMate. Consider for example the value object 'Tailored car insurance package guarantee' that the customer receives from the intermediary in the intermediary value web (see Fig. 2). One models this object in e^3*value* because it reflects one of the primary reasons for a customer to consider an intermediary valuable: to find a tailored insurance package that matches his needs. An ArchiMate model, however, would typically not contain this value object because it has no *tangible* impact on the operationalization of the underlying value web (In terms of business processes and information systems).

On the other hand, a business object in ArchiMate does not necessarily translate to a value object in e^3*value*. An invoice is a good example of this: in the Archinsurance case, an invoice may be modelled because it is relevant for the underlying IT implementation of the Archinsurance value web. However, invoices are often operational objects, and therefore often do not show up in a value model.

Value activity concept. The concept of a value activity in e^3*value* differs from the concept of a business function in ArchiMate. In the Archinsurance case, we could easily transition from a e^3*value* value activity, such as 'Contracting', to a similarly named business function in ArchiMate. This is because the notion of a value activity is inclusive to the more broadly defined business function concept from ArchiMate, the latter being used to "represent what is most stable about a company in terms of the primary activities it performs“ [9, p. 24].

Thus, also here we need to account for the fact that an e^3*value* value activity maintains value as a main criterion, whereas an ArchiMate business function does not.

Economic reciprocity. As stated in Sect. 2.1, the idea of economic reciprocity is central to e^3*value*. As modelled by the e^3*value* concept of a value interface, actors only offer an object of value to another actor, if they receive compensation in return. For example: in the Archinsurance intermediary value web (Fig. 3), we see that an intermediary offers a 'Complete risk profile' to Archinsurance, and receives the value object 'Premium' as compensation.

ArchiMate, however, does not have a concept to depict economic reciprocity (again, because of its operational nature). So, for example, in ArchiMate one cannot express that a 'Complete risk profile' is only provided by the intermediary to Archinsurance if a 'Premium' is received in return.

Mapping limitations. We find commonalities between ArchiMate and e^3*value* that allow us to map between these models. In particular, we can map between the e^3*value* concepts 'Actor', 'Market segment' and 'Value activity' on the one hand, and the ArchiMate concepts 'Actor' and 'Business function' on the other.

However, because of the operational, respectively value focus of ArchiMate and e^3*value* we find differences between concepts from the aforementioned models (as shown in this section). In particular, we find differences in: *the actor concept*, which in e^3*value* has a profit-and-loss responsible criterion but not in ArchiMate, *the service concept*, which e^3*value* considers as a commercial service, whereas ArchiMate considers services both on a commercial and transactional level, the *value object* concept, which is usually not modelled in ArchiMate because of ArchiMate's operational nature, *the value activity concept*, which in e^3*value* usually has an external focus (i.e., it is modelled in as far it is relevant in a network of enterprises) whereas the business function in ArchiMate has more of internal focus (ie the focus is on modelling activities within an organisation), and, finally, ArchiMate lacks the notion of economic reciprocity.

5 Related Work

The e^3*alignment* approach provides tools for actually creating business-ICT alignment. It does so by ensuring that conceptual models depicting a strategic, value, process and ICT perspectives respectively on the value web at hand are consistent with one another [14]. However, this approach works only on a syntactic level. For instance, if the concept of an actor in e^3*value* and the concept of a swim lane in an UML activity diagram actually means the same is not a consideration. Derzsi et al. enable profitability calculations of an ICT-infrastructure by providing a meta-model that links an IT infrastructure modelled in UML to e^3*value* [15]. This approach has more formality than e^3*alignment* , yet it focuses on a link between IT and value only. As a result, business processes are not a consideration while these are realistically cost carriers as well.

The Object Management Group (OMG) presented the Unified Modelling Language (UML) (version 2.0) [16], standardized in 2004, which is the backbone of the object oriented software engineering computing paradigm. UML offers comprehensive support for control-flow and data perspectives [17]. However UML techniques provide limited support for modelling organisational or value aspects of business processes. These limitations are common to many other business process modelling formalisms such as the Business Process Modelling Notation (BPMN) and reflect the emphasis that has been placed on the control-flow and data perspectives in contemporary modelling notations [18].

Ontological merging approaches address the semi-automated integration of system models [19]. System models are created in terms of modelling language, which in itself is based on a meta-model. Syntactic and semantic mapping between pairs of meta-models has been facilitated by the application of existing approaches for ontology mapping [20, 21]. Ontologies improve not only the semantics of a meta model but also provide a potential way in which these meta models can be bridged with each other to be integrated within a common context [22]. However, ontology mapping approaches such as [20, 21] focus on providing an approximation of a mapping between two ontologies. Yet, in our research we require a precise mapping. Since our starting point are ontologies, such as e^3*value*, with relatively few concepts (compared to larger ones such as found in the medical domain), it seems better to perform mapping/integration manually and as such, avoid an approximation of a mapping.

6 Conclusion and Future Directions

In this paper, we provided a step-wise, intuitive, mapping approach for integrating the value modelling technique e^3*value* into the enterprise architecture framework ArchiMate. Following up on this, we discussed the limitations of our step-wise mapping approach as an input for further research. Also, we showed why integration of e^3*value* and ArchiMate is useful. On the one hand, e^3*value* is complementary to ArchiMate in terms of profitability calculations while, one the other hand, ArchiMate is complementary to e^3*value* in terms of operationalizing a proposed business collaboration.

Our goal is to ensure the consistency of different modelling techniques and to foster the notion of model traceability. For future research, we therefore intend to explore techniques that can be used to relate conceptual models beyond naive concept mapping. One such technique is the fact-based modelling technique Object Role Modelling (ORM) [11]. The basic idea behind applying ORM is to create an initial meta model based on the concepts present in ArchiMate, while expanding and adjusting it based on concepts from other ontologies.

Acknowledgements This paper results from efforts carried out in the projects ASINE and Agile Service Development (ASD).
The ASINE project (http://asine.tudor.lu), from CRP Henri Tudor, is funded by the FNR program 'PEARL'. The ASD project (http://asd.novay.nl) is part of the program Service Innovation ICT of the Dutch Ministry of Economic Affairs, Agriculture and Innovation.

References

1. D. Tapscott, D. Ticoll, and L. A., *Digital Capital: Harnessing the Power of Business Webs*. Boston, Massachusetts: Harvard Business Press, 2000.
2. M. Lankhorst *et al.*, *Enterprise Architecture at Work: Modelling, Communication and Analysis*. Springer, Berlin, Germany, 2005.
3. M. Op 't Land, H. Proper, M. Waage, J. Cloo, and C. Steghuis, *Enterprise Architecture – Creating Value by Informed Governance*. Springer, Berlin, Germany, 2008.
4. H. Jonkers, H. Proper, and M. Turner, "TOGAF and ArchiMate: A Future Together," The Open Group, White Paper W192, November 2009. [Online]. Available: http://www.opengroup.org/bookstore/catalog/w192.htm
5. R. v. Buuren, J. Gordijn, and W. Janssen, "Business case modelling for e-services," in *18 th Bled eConference eIntegration in Action*, 2005.
6. B. Nuseibeh, J. Kramer, and A. Finkelstein, "A framework for expressing the relationships between multiple views in requirements specification," *Software Engineering, IEEE Transactions on*, vol. 20, no. 10, pp. 760–773, 1994.
7. J. Cummins and N. Doherty, "The economics of insurance intermediaries," *The Journal of Risk and Insurance*, vol. 73, no. 3, pp. 359–396, 2006.
8. M. Lankhorst *et al.*, "ArchiMate Language Primer," *Telematica institute*, 2004.
9. M. Lankhorst, "Viewpoints Functionality and Examples," *Telematica Institute*, 2004.
10. P. Bernus, L. Nemes, and G. Schmidt, Eds., *Handbook on Enterprise Architecture*, ser. International Handbooks on Information Systems. Springer, Berlin, Germany, 2003.
11. M. Lankhorst, H. Proper, and H. Jonkers, "The Architecture of the ArchiMate Language," *Enterprise, Business-Process and Information Systems Modeling*, pp. 367–380, 2009.

12. J. Gordijn, P. De Leenheer, and I. Razo-Zapata, "Generating service valuewebs by hierarchical configuration: An ipr case," in *Proceedings of HICSS 44*, 2011.
13. J. Gordijn and H. Akkermans, "Value based requirements engineering: Exploring innovative e-commerce ideas," *Requirements Engineering Journal*, vol. 8, no. 2, pp. 114–134, 2003.
14. V. Pijpers, J. Gordijn, and H. Akkermans, "e3alignment: Exploring inter-organizational alignment in networked value constellations," *International Journal of Computer Science & Applications*, p. 59, 2009.
15. Z. Derzsi, J. Gordijn, and K. Kok, "Multi-perspective assessment of scalability of it-enabled networked constellations," in *Proceedings of the 41st Annual Hawaii International Conference on System Sciences*, R. H. Sprague, Ed. IEEE CS, 2008, p. 492.
16. O. M. Group, "Uml 2.0 superstructure specification, available at: http://www.omg.org/cgibin/ doc?ptc/2004-10-02." 2004.
17. N. Russell, W. M. P. v. d. Aalst, A. H. M. t. Hofstede, and P. Wohed, "On the suitability of uml 2.0 activity diagrams for business process modelling," in *APCCM '06: Proceedings of the 3rd Asia-Pacific conference on Conceptual modelling.* Darlinghurst, Australia, Australia: Australian Computer Society, Inc., 2006, pp. 95–104.
18. R. K. L. Ko, S. S. G. Lee, and E. W. Lee, "Business process management (bpm) standards: A survey," in *Business Process Management journal Vol. 15 No. 5, 2009. Emerald Group Publishing Limited. Accepted on 2 Dec 2008*, 2009.
19. V. Devedzić, "Understanding ontological engineering," *Commun. ACM*, vol. 45, pp. 136–144, April 2002. [Online]. Available: http://doi.acm.org/10.1145/505248.506002
20. M. Ehrig and S. Staab, "Qom–quick ontology mapping," *The Semantic Web–ISWC 2004*, pp. 683–697, 2004.
21. N. Noy and M. Musen, "The prompt suite: interactive tools for ontology merging and mapping," *International Journal of Human-Computer Studies*, vol. 59, no. 6, pp. 983–1024, 2003.
22. H. Happel and S. Seedorf, "Applications of ontologies in software engineering," in *In 2nd International Workshop on Semantic Web Enabled Software Engineering (SWESE 2006), held at the 5th International Semantic Web Conference (ISWC 2006)*, 2006.