

Value-based Process Model Design

Felicia Hotie, Jaap Gordijn

Received: date / Accepted: date

Abstract Networked service value constellations, consisting of enterprises and customers working together to jointly provide a commercial service, can be analyzed from different modeling perspectives. Two of such perspectives are (1) the value perspective and (2) the process perspective. Value models, describing the value perspective, indicate which economically valuable service outcomes are exchanged between the involved actors. However, a value model only shows *what* is exchanged of economic value, but not *how* this should be accomplished. Therefore, to understand a service well, a process model has to be designed also, which shows the actual tasks to be performed by these actors as well as the time ordering of these tasks. A key problem is then how to construct such a process model, given an earlier designed value model. As the process model should put the value model into operation, there exists a clear relationship between both models. Previous work investigated this problem to a certain extent, but a well integrated, and easy to use method is currently lacking. This paper proposes a step-wise method to design a process model for networked value constellations, given an earlier developed value model. The aim of this method is to support practitioners during the design of a process model; as a result, the proposed way of working should tractable, well teachable, and easy to use, thereby following the same philosophy as with the *e³value* methodology, which we use to model value perspective of networked value constellations. In addition to the step-wise method itself, the value of this paper is also in the use of the method to explore services related to Intellectual Property Rights (IPR) clearing.

Keywords Value model, Process model, e³value, BPMN, Model construction

1 Introduction

Commercial services are an important economic factor. In Europe, the services sector is estimated to be 71.3% of the GDP in 2015 ¹. Many services are in fact *electronic services*. These services are ordered and provisioned via the Internet [Mohan & Ramesh, 2003]. Examples include e-banking, software-as-a-service and electronic Intellectual Property Right (IPR) clearing, the domain of our business partner in this research.

Commercial services have *economic value* for the customer, therefore the customer has to pay a certain amount of money (or another compensation) to the supplier(s) of the service. Additionally, services suppose a process, carried out by the supplier(s) and sometimes even by the customer. For example, a meal at a self-service restaurant requires an activity performed by the cook *and* by the customer.

A commercial service assumes a network of participating actors. The network contains at least one supplier and a customer, but in more complex services, many suppliers are involved to satisfy a customer need. We call such a network of actors a *value constellation* [Normann & Ramírez, 1994, Normann & Ramírez, 1993].

For electronic commercial services (e.g. Spotify, Netflix, etc.), which heavily rely on information technology, the accepted way of working is to produce requirement specifications and designs in terms of *conceptual models* [Mylopoulos et al., 1990] (e.g. UML diagrams). We consider *conceptual modeling* as a useful way of working; not only to understand the required information systems and business processes, but also to design and evaluate the service value proposition of a constellation of enterprises and end users in terms of a *value model*. Therefore, our long term goal is to contribute a model-based design method supporting the development of commercial networked service value constellations, starting at the value proposition to the customer and ending with the enabling information technology.

This paper considers two specific modeling techniques: *e³value* [Gordijn & Akkermans, 2001] for understanding such service value constellations, and the quasi standard for business process modeling, the Business Process Modeling Notation (BPMN) [Owen & Raj, 2003, OMG, 2011] for designing business processes for constellations. Note that we have chosen BPMN for pragmatic reasons; since our findings are applicable to a wide range process modeling techniques, our contribution is agnostic with respect to the selected process modeling technique. The proposed method is also useful for value modeling techniques similar to *e³value*, such as the Resource Event Agent (REA) ontology [Geerts & McCarthy, 1999].

Although modeling different concerns, the *e³value* models and the BPMN models are strongly related. The *e³value* model represents actors (suppliers and customers) exchanging things of economical value (service outcomes and

¹ see <https://www.cia.gov/library/publications/the-world-factbook/geos/ee.html>, visited 24 september 2016

money) with each other. The concern here is *profitability*. Profitability is assessed by calculating the net cash flow for all actors involved. In order for a value constellation to be sustainable, *all* actors in the constellation should be able to generate a *positive* net cash flow. In contrast, the business process model shows *how* the value model is put into operation, in terms of activities, their time ordering, and messages exchanged. The concern here is the *control flow* of operational activities, as well as the *assignment* of these activities to actors.

A key problem while using two different modeling languages to understand different aspects of a service under study is that potentially these models may overlap, or relate in a different way, thereby exposing consistency issues. The formal consistency relationship between *e³value* and process modeling has been investigated by e.g. [Bodenstaff, 2010]. Another point of departure with respect to relating different perspectives on the same artifact is to consider how to *derive* the one perspective, given the other perspective. In our case, this implies deriving the BPMN diagram, given an *e³value* diagram. However, the current state of the art with respect to deriving a process model based on a value model is rather fragmented, and a well integrated method, usable by practitioners, is lacking.

This paper proposes an easy to understand, and integrated, step-wise method to develop a process model given a value model, bringing together earlier developed work by a number of researchers. The aim of this method is simplicity of use; which is the same philosophy behind the *e³value* methodology. The method should be lightweight, easy to understand, and teachable, as service development projects are typically characterized by short time-to-market. We assume that the user of our method is a business development consultant with conceptual modeling skills.

To develop and validate the step-wise method, we employ the Technical Action Research (TAR) [Wieringa, 2014] method. The TAR method (1) identifies a problem to be solved by a treatment (here: how to design a process model given a value model), (2) proposes a treatment (here: the step-wise method and (intermediate) models), (3) applies the treatment in a real-life context (here: Intellectual Property Rights (IPR) clearing in the music industry), and (4) reflects on the treatment in the real-life context, with the aim of learning and improving.

This paper is structured as follows. First, in section 2, we review the existing work on relating value models and process models, as we use this work to compose an integrated and easy to learn step-wise method to develop value-based process models. Section 3, introduces *e³value*, the technique we use to represent value models. For process models we use BPMN, and we assume the reader is familiar with this technique. In section 4 we present the TAR method. Hereafter, we discuss the step-wise method for deriving a process model based on a value model. Then, we apply the proposed method to our IPR business partner. In section 7, we reflect on the use of our method and suggest changes and improvements. Finally, the conclusion and suggestions for future work are presented.

2 Designing Process Models Using Value Models

During commercial service development in value constellations, a number of activities have to be done, and two of them are (1) developing the value propositions (what are we actually offering to whom, and what do we request in return), and (2) designing a process that provisions the propositions (as many services are in fact processes in terms of provisioning).

The traditional way of doing business development is rather verbose, with ideas expressed in natural language. Using natural language as requirements specification language exposes a number of well known drawbacks such as noise (irrelevant information), silence (omission of important information), overspecification, contradictions, ambiguity, forward references, and wishful thinking [Meyer, 1985].

A conceptual model-based method would mitigate these problems, and additionally, if the method allows for a graphical specification, communication with stakeholders is easier [Wiegers, 1999]. Moreover, conceptual models allows for semi-automated analysis. For example, an *e³value* model (see Sec. 3 for a brief introduction) can be analyzed for a positive net cash flow for all actors involved, and a BPMN process specification can provide the groundwork for a simulation of the process at hand. Last, but not least, we argue that business development should start with a business *value model*, with an emphasis on *model*, and *economic value*. In many cases, business development starts with business process design immediately, thereby implicitly assuming value propositions. We claim that value proposition design should be a first class citizen in the design of service value constellations, allowing to assess promising propositions, and selecting the most interesting ones.

Taking two perspectives on a service constellation (namely a business value and business process perspective) carries the risk that both perspectives appear to be unrelated, or even inconsistent with each other. To relate process models and value models, two view points can be taken: (1) considering consistency between both models explicitly, and (2) designing a process model given a value model, such that the process model puts the value model into operation.

Understanding consistency between both model types allows to do (formal) model checking between a particular value model and a process model. In [Bodenstaff et al., 2007, Bodenstaff, 2010], a framework is proposed to check and maintain consistency between a value model and the corresponding process oriented coordination model. The framework allows checking business and coordination models on a structural model level, that is the specification of the value- and process models, but also on an instance level, namely the execution of the process in the terms of a workflow management system. The notion of value model and process model *instance* deserves further explanation. Normally, an *e³value* model describes, for a particular time period (e.g. a year), the net value flow. Using profitability analysis, net value flow sheets can be generated. These sheets show the expected profit, based on a number of variables (e.g. number of customers, pricing formulas, etc.), and are considered as a running instance of the value model. During the same period, the process

instance runs, resulting in actors exchanging objects of value with each other. The economic results of a running process instance should ideally, for a chosen time period, match with the expected profits from the value model. Although consistency checking between value and process models provides some starting points for deriving process models from value models (e.g. by considering the consistency rules themselves), it is not an explicit *design* method, which helps consultants to develop a process model for the corresponding value model. In fact, it supposes existence of both models already, whereas the design problem is to find a suitable process model that implements a value model.

In [Schuster et al., 2010, Schuster & Motal, 2009], a semi-formal mapping is proposed between the *e³value* methodology, REA [Geerts & McCarthy, 1999], and UMM [Huemer et al., 2011]. This method gives mapping rules, but acknowledges that sometimes mapping requires manual work.

Therefore, in this paper, we argue that creating a process model based on a value model should not be seen as a translation or (automated) mapping problem between both models only. The semantic gap between models is too large to allow for a mapping-only approach. Therefore, in our proposal, we consider the transition from a value model to a process model explicitly as a *human* design process.

Based on an analysis of previous work [Pijpers & Gordijn, 2008a, Pijpers & Gordijn, 2007a, Weigand et al., 2006, Weigand et al., 2007b, Wieringa & Gordijn, 2005] we conclude that to transit from a value model to a process model, the following two questions should be answered:

- Do actors trust each other? (as a value model assumes a perfectly honest world)
- Is the physical possession flow of objects different than the ownership flow of objects? (as a value model only considers flow of ownership)

Trust motivations can influence the time ordering of value transfers, and therefore are of relevance to process models. For instance, a shop may require a payment in advance, before the shop actually delivers the products. We have studied trust issues extensively, see for instance [Kartseva et al., 2009], and more recently [Ionita et al., 2015]. In order to represent solutions with respect to trust, we change the *e³value* notation such that time ordering of value transfers can be represented. We call the resulting model an *e³value trust* model, as the model still only represents transfer of economic valuable objects, and not topics such as message flows and control flows, which are common in process models.

The other important decision towards a process is the physical flow of value objects, and more specifically the distinction between the *possession* flow and *ownership* flow of objects. In the *e³value* methodology, we are only interested in ownership, as a value transfer implies a transfer of ownership of goods, or the transfer of the right to enjoy an experience in case of intangible services. However, in order to transfer ownership, a different physical possession flow of objects may happen. For instance, if we order a book at a web shop, a logistic partner will be involved. That logistic partner has physical possession of the

book for some time, but does not own that book. The distinction between ownership and possession of an object has been acknowledged by a number of authors. In [Weigand et al., 2007a], a value object is considered as a value transformation and the right transfer on a resource, whereas for the process perspective, the physical transfer is identified, which comes close to our notion of the physical possession flow of objects. We use the distinction between ownership and possession rights as an aid to develop process models in [Pijpers & Gordijn, 2007b], in a method called *e³transition*. Here, ownership is defined “as the right to use and claim possession of the value object” (see also [Snijders & Rank-Berenschot, 2001]) [Pijpers & Gordijn, 2007b]. Also possession is identified as “the right to have actual (and if possible physical) possession of an object [Snijders & Rank-Berenschot, 2001], but not to use the object” [Pijpers & Gordijn, 2007b]. We use both definitions extensively in our work. The physical flow of value objects is represented by an *e³value possession* flow model.

Since the *e³value* trust and possession flow models are just slight variations of the original *e³value* model, these variants are easy to learn, assuming that the modeler already is experienced in designing *e³value* models.

3 The *e³value* technique

We briefly explain the *e³value* technique using an educational example (see Fig. 1); for a more detailed overview the reader is referred to [Gordijn & Akkermans, 2001, Gordijn & Akkermans, 2003].

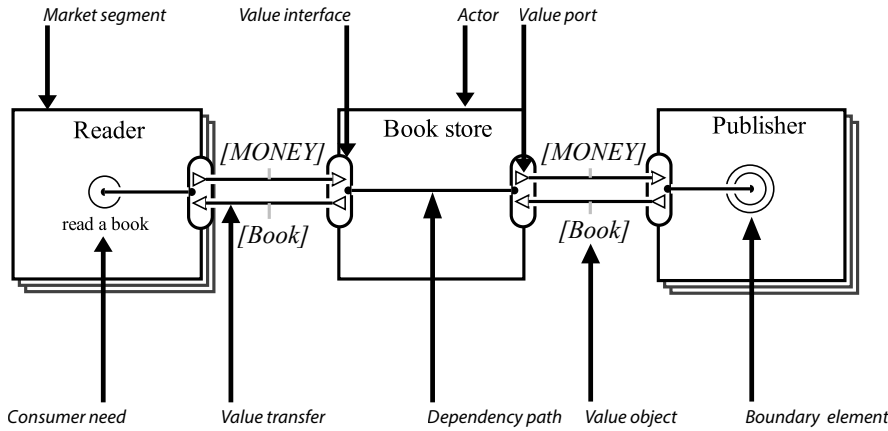


Fig. 1: An educational *e³value* model

Legal, profit-and-loss responsible, entities (both end-users and enterprises) are represented as *actors* and *market segments*. An actor is a single end-user or enterprise; a market segment is a set of actors, who economically value things

equally, and is used to state that multiple actors of the same kind exist. In the example, there are many readers who want to read a book, precisely one book store, and multiple publishers.

Actors offer and request *value objects* to and from their environment. A value object is for at least one actor of economic value. In the example, value objects are: book, and money. Actors request and offer value objects via *value ports*, thereby abstracting away how these objects are delivered, which is the focus of business *process* design. Ports are grouped into *value interfaces*, which denote economic reciprocity and atomicity. Therefore, in the model, it is only possible to obtain a book while paying for it with money, and vice versa. Note that this supposes an ideal, honest world; in other words, actors do not commit a fraud. The focus of an e^3 value model is how value is offered, requested, and transferred in a network, and not on all kinds of deviations.

Value ports of actors are connected by means of *value transfers*. These represent that actors exchange value objects, in the example books and money.

Finally, there is the notion of *dependency path*. Such a path starts with a *consumer need*, here the need to read a book. The actor (the reader) needs to exchange value objects via its value interface to satisfy such a need, in the example a book for money. The book store needs to exchange value objects too, to deliver the book to the reader. In fact, the book store obtains the book from the publisher. The publisher has a *boundary element*, signaling that no further value transfers are considered. When to use a boundary element is a modeling decision. It could be argued that the publishers need paper and ink, to print the book; if the modeler wants to express this, additional transfers would be needed.

An e^3 value model can be quantified, e.g. by stating the number of actors in a market segment, the number of consumer needs, pricing, etc. Tool support can then calculate the number of transfers, and the net cash flow for each actor involved.

There exist two other ontological founded approaches for value-oriented business modeling: the Business Model Canvas [Osterwalder & Pigneur, 2010] (BMC), and the Resource Event Agent (REA) ontology [Geerts & McCarthy, 1999]. BMC has a single actor perspective, whereas e^3 value and REA have a multi-actor perspective. Also, BMC is informal, and therefore can not do tool supported analysis, specifically net cash flow calculation. REA and e^3 value are quite similar although they have a different origin. REA was developed as an ontology to represent the accounting domain, whereas e^3 value was exclusively designed for business development. Ontologically, e^3 value is richer in its constructs, for instance to represent dependencies between actors (dependency path), actor composition (actors having joint value propositions), multi-party (> 2 actors) transactions, and value activities.

4 Technical Action Research as a Research Instrument for Design Science

In [Hevner et al., 2004], Design Science is framed as “the scientific study and creation of artifacts as they are developed and used by people with the goal of solving problems of general interest.” This paper addresses the problem of finding a suitable process model given a value model for networked value constellations. These artifacts can be *methods* (in our case the way how to develop process models based on value models) as well as *models* (the (intermediate) models, from value model to ultimately a process model).

There exist multiple research strategies to study the above mentioned artifacts. As argued in Sec. 2, already a considerable amount of ground work has been done on designing process models by using value models. However, this work is fragmented, and so what misses is an integrated method to make these works really usable for practitioners. The aim of this research therefore is (1) to develop an easy to understand, step-wise method for deriving process models from value models, which can be followed by practitioners, and (2) to test this method in a real world setting.

To learn about a method *in practice*, [Wieringa, 2014] proposes the Technical Action Research (TAR) method. TAR specifically focuses on the *artifacts* (developing process models using value models, and the models themselves), rather than on the *problems* in general, as many other research methods do. Therefore, TAR fits our research goals, namely to develop an method to derive process models from models (the artifacts), and doing so in a real-life context. The TAR cycle comprises (1) problem statement, to be solved by a treatment, (2) treatment design, (3) usage of the treatment in a real-life context, and (4) evaluate and improve the treatment.

The problem this paper aims to solve is how to design a process model for networked value constellations, given a value model as input. The goal of the method is that it is (1) model based, cf. the accepted way of working in Requirements Engineering, and (2) usable by practitioners, having modeling skills (e.g. the UML). We assume that the practitioner is capable of designing e^3 value networked business model himself; in other words, our method is not a recipe for developing e^3 value models.

The TAR method is in our paper further used as follows. Section 5 presents our treatment design, which is the step-wise method to derive a process model based on a given value model, as well as the required intermediate models. In section 6, we apply the proposed treatment to a service value constellation whose activity comprises clearing of intellectual property rights. Finally, section 7 evaluates the treatment and suggests improvements to the step-wise method.

For the real-life context, we have selected a company that works in a networked value constellation since value models are primarily intended for designing networked business models. Our selected business partner (hereafter called the IPR society) clears IPR on music for music users (radio and television, restaurants, etc.), pays right holders (artists, producers) for use of their

music, and uses information from radio stations for doing so. Since a number of different actors are involved in this process of rights clearing, these actors form a value constellation by definition.

Regarding validation and evaluation of our method, we have deliberately chosen for the *known* process of clearing international IPR. The same holds for the value model. This allows us to evaluate whether our method produces a process model that is close to the known reality.

5 Treatment Design: A Step-wise Method to Design a Process Model Using a Value Model

The aim of this paper is to develop a step-wise method for the design of a process model using a value model for networked value constellations, making use of the existing research related to this problem (see Sec. 2). As explained, two different design decisions frequently occur in the literature:

- How to deal with (lack of) trust between actors in a networked value constellation?
- What is the *physical* flow of objects in a process executed, as opposed to the *ownership* flow of the same object in the value model?

Our proposed treatment will be based on supporting the above mentioned design decisions. We design our treatment, the step-wise method, using a small, easy to understand, example. The example networked value constellation consists of Customers, a Seller, a Carrier for handling logistics, and a Bank. The Seller is a web shop, so needs a logistic provider to transport the product to the Customer. The Customer and Seller exchange a product (e.g. a book) for money, the Carrier provides a transportation service to the Seller and gets paid for this, and the Bank provides payment services to the Customers and the Sellers and gets paid for service provisioning. The corresponding e^3 value model is shown in Fig. 2.

We have carefully designed this example such that the example exhibits the earlier identified two design decisions while developing a process model given a value model:

- Trust relations may vary. For example, the Carrier may trust the Seller, since they do business on regular basis, but the Seller may not trust the Customer, because they interact only incidentally.
- The ownership flow of objects (product ownership flows from Seller to Customer) in the e^3 value model is expected to be different than the related physical possession flow in the process model (products possession goes from Seller to Carrier, and then from Carrier to Customer).

The naive approach would be to, preferably automatically, translate the value model for the example at hand into a process model. However, we argue that a value model cannot be *directly* translated into a process model because many different intermediate design decisions are of relevance [Pijpers

& Gordijn, 2007b][Pijpers & Gordijn, 2008b]. For instance, the *time ordering* of the value transfers and the physical flow of the value objects are very relevant to process models, but are simply not given in a value model, as the value model shows only *dependencies* between value transfers, without stating whether the dependee should occur before or after the dependent.

To arrive at a process model, given a value model, we therefore consider trust issues between actors, as well as physical flow of value objects between actors. We represent these two considerations by means of two intermediate models, namely the *trust model* and the *possession flow model*.

To keep the complexity of our step-wise method to the bare minimum, the models used to represent solutions, to trust issues and the physical value flow of objects, are models that are slight, and precise articulated variations of the e^3 value models. As we assume that practitioners are already capable of developing e^3 value models, we therefore expect that these practitioners can easily learn the required variations in the e^3 value modeling language to deal with trust and possession considerations.

In sum, to create a process model, we propose to perform the following three steps: (1) create the trust model based on the initial value model, (2) create the possession flow model based on the trust model, and (3) create the process model based on the possession flow model. In the remainder of this section we discuss this step-wise method in more detail by means of the web shop example.

5.1 Value Model

Our step-wise method supposes a value model as starting point. Construction of such a model is outside the scope of this work; instead we assume a suitable modeling language and methodology to construct such a value model. We use the e^3 value modeling language [Gordijn & Akkermans, 2001] for expressing a networked value model.

Figure 2 shows the value model example about a web shop with the following four economically rational actors: (1) the Customer, (2) the Seller, (3) the Carrier and (4) the Bank. As there are multiple customers, the actor Customer is depicted as a market segment. In the e^3 value methodology, a market segment refers to a set of actors that are supposed to value things equally. The Seller may obtain the products from others, which however we consider as not in the scope in this model.

If a Customer obtains a good from the Seller, both the Customer and Seller need a paying instrument (e.g. an online bank account). This is indicated by *note #1.1* and *#1.2* in Figure 2: It represents that in order to satisfy a consumer need, the Customer must *both* obtain a product and a *payment service*. Also, the Seller uses a payment service in order to obtain payments. Note that the value model does *not* show that money is actually flowing from the Customer, *via* a Bank, to the Seller. This will be visible in the possession flow model (see Sec. 5.3). Instead the value model shows that (1) the Customer

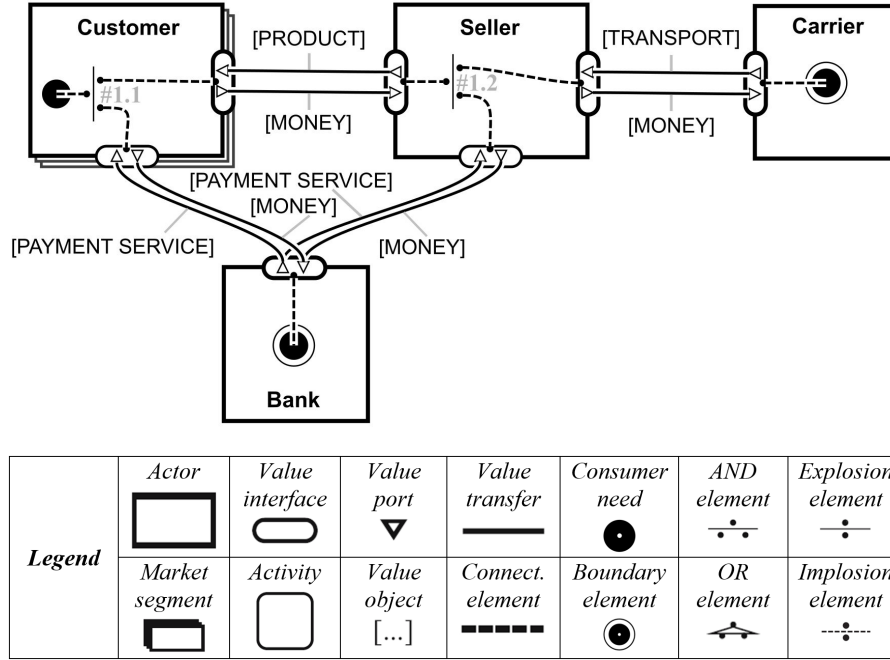


Fig. 2: Value model - Web shop

pays the Seller (for obtaining a product), and in order to do so, (2) a payment service is required, and that the Customer and Seller pay for such a service.

Similarly, the Seller needs to obtain a transportation service from the Carrier, and pays for this service. Therefore, from a logical value point of view, the Seller transfers the product to the Customer. This is despite the fact that the Carrier *physically* transports the product to the Customer. The reason for this is that a value model only models the *value* transfers in terms of *ownership* between the actors. The Carrier is only an intermediate actor for the Customer and there are no value transfers in terms of ownership between the Customer and the Carrier.

A key notion in e^3value is the *dependency path*. This path, consisting of consumer needs, value interfaces, value transfers, connection elements, and boundary elements, shows which value transfers must happen, as a result of the occurrence of a consumer need. In Figure 2, by considering the need, it can be seen that a product is exchanged for money, a payment service is exchanged for a payment (both for the Customer and the Seller), and that transportation is exchanged for money.

The e^3value modeling language has a number of restrictions. Firstly, the *time ordering* of the value transfers of the involved actors on a path is not represented in an e^3value model. The model only shows that for a good, a payment is received, but not if such a payment happens before or after deliv-

ering the good. The focus is only on understanding which value transfers must happen to satisfy a consumer need, which is already sufficiently challenging in business development projects. Secondly, an e^3value model does not allow connection elements to be located ‘outside’ the actors. Connection elements show the value objects the actor must exchange, as a result of other exchanges of that actor, e.g. to show that a sold product results in the need for a transportation service. Thirdly, an e^3value model does not show the actual physical flow of the objects. Only the ownership right of the value objects are transferred in an e^3value model. Therefore, the physical flow of a product from the Carrier to the Customer is not shown in an e^3value model. Lastly, it is not possible to aggregate value transfers (e.g. payments) in an e^3value model. For instance, the model states that for each use of the transportation service a certain amount of money has to be paid. What might physically happen however, is that the Seller pays the Carrier the indebted amount of money on a monthly basis, which is aggregated amount of money for each transportation service in that month. This is not represented in an e^3value model. In order to take into account these limitations, we create two follow-up models.

5.2 From Value Model to Trust Model

The first intermediate model is a so-called trust model. We create a trust model inspired by the e^3value modeling language, but with a more extensive use of Use Case Maps by Buhr (1998) [Buhr, 1998]. A trust model is based on the initial value model, but now shows the *time ordering* of the value transfers. The value objects are transferred following a particular time ordering, and in which order this happens depends on the trust motivations and considerations between the participating actors. The time ordering of value transfers varies by each organization and situation. For example, in general a customer pays in advance when purchasing a product in a web shop. However, there exist web shops that included the ‘buy now, pay later’ option [Mendoza & Pracejus, 1997]. This allows customers to pay after receiving the product(s), but requires a deposit upfront.

We assume that the actors, the value ports, the value interfaces, the value objects and the value transfers are already properly identified in a value model. Thus, we adopt these model concepts from the value model to the trust model. It is important to understand that the value transfers remain the same, but a trust model also shows the time ordering of these value transfers. To do so, the connection elements of a dependency path do not connect to value interfaces (as in an e^3value model), but connect to the value ports inside a value interface. Because value ports relate value transfers, it is now possible to model the time ordering of the transfers. Additionally, a trust model allow connection elements to be located ‘outside’ the actors. This is in contrast with the e^3value modeling language that only allows to use connection elements within the same actor.

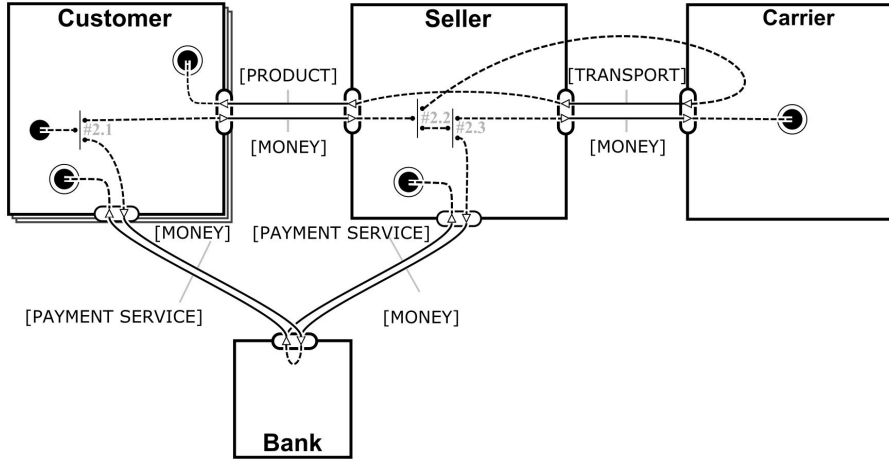


Fig. 3: Trust model - Web shop

We adopt the same consumer need, actors and value transfers from Figure 2 to Figure 3. In this specific Web shop example we assume that the Customer pays the Seller at the point of sale. To perform this payment transaction, a payment service from the Bank is needed. Therefore the Customer pays the Seller and the Bank simultaneously (*see note #2.1*). In return, the Bank provides a payment service to the Customer. Once the Seller received the money from the Customer, the Seller obtains a transportation service from the Carrier and therefore has to pay the Carrier in return (*see note #2.2*). To make the payment, the Seller pays the Carrier and the Bank concurrently (*see note #2.3*). The Bank provides a payment service to the Seller in return. After the Carrier provided the transportation service, the Seller provides the product to the Customer. Again, there is no direct transfer between the Carrier and the Customer since a trust model also represents only the *value* transfers in terms of ownership between the actors. Note that additional dependency paths, AND-forks and boundary elements are modeled in Figure 3 to represent the correct time ordering of the value transfers.

5.3 From Trust Model to Possession Flow Model

The following intermediate step is to create a possession flow model using the already created trust model. A possession flow model represents the physical flow of objects and is inspired by the *e³transition* model by Pijpers and Gordijn (2007) [Pijpers & Gordijn, 2007b]. The *e³transition* model shows the independent transfers of objects and their possession rights. It is possible that a value object is transferred by a certain actor, but that this actor does not have the *ownership* over this object [Pijpers & Gordijn, 2007b]. For example, a Carrier possesses a product and has to transport this product from the Seller

to the Customer, but the Carrier is not the *owner* of the product. This transfer is called the *possession transfer* and is not considered as an *economic value transfer*, hence these possession transfers are not included in a value model and a trust model. However, these possession transfers are represented in a possession flow model. Furthermore, a possession flow model enables to aggregate a number of similar value transfers because of efficiency reasons, e.g. reducing the amount of transactions and/or transaction costs. For instance, all payments for a particular actor can be aggregated for a month and dealt with as one. The possession flow model of the Web shop example is presented in Figure 4. We adopt the same actors from Figure 3 to Figure 4. Note that the actor Bank is repeated twice to keep the possession flow model less cluttered in terms of model layout. In contrast to Figure 3, we also add the possession transfers to Figure 4.

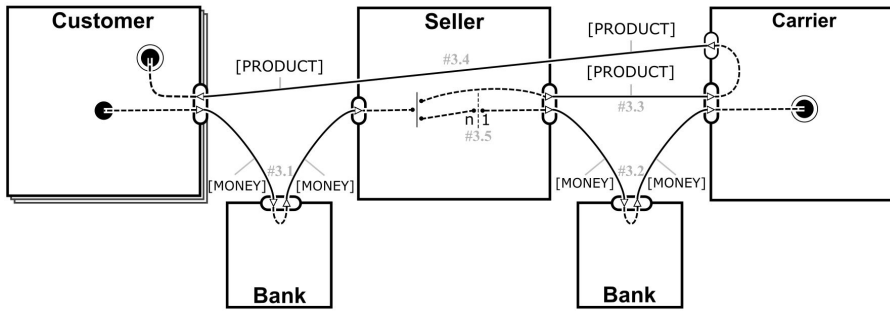


Fig. 4: Possession flow model - Web shop

In Figure 4 two money possession transfers are added. The first money possession transfer goes from the Customer to the Bank and from the Bank to the Seller (see note #3.1). The Customer pays the Bank per transaction, thus the Bank keeps the transaction fee and then sends the remaining agreed amount of money to the Seller. The money possession transfer between the Seller and the Carrier proceeds via the Bank as well (see note #3.2). Also, the Seller pays per transaction to the Bank. Note that the value transfer ‘payment service’ is omitted twice in Figure 4. The money possession transfers replace this value transfer because the money possession transfers already implicitly indicate that the Bank provides a payment service to the Customer and the Seller. This way, we avoid superfluous transfers between the actors.

In addition, two product possession transfers are depicted in Figure 4. The first product possession transfer shows that the product is transferred to the Carrier (see note #3.3). In this model we do not take into consideration whether the Carrier picks up the product at the Seller or whether the Seller hands over the product to the Carrier. Note that this product possession transfer replaces the ‘transport’ value transfer modeled in Figure 3. The next product possession transfer (see note #3.4) already implicitly indicates that

the Carrier provides a transportation service to the Seller by showing that the Carrier delivers the ordered product to the Customer. Thus, it is unnecessary to model the value transfer ‘transport’ in the possession flow model. The Carrier possesses the value object at the moment, but does not have the ownership over the product. The Carrier is only responsible for transporting the product from the Seller to the Customer. The Customer is the actual owner of the product.

Finally, we model an aggregation of payments using an implosion element (*see note #3.5*). The Seller pays the Carrier for multiple value transfers, in this case product transportation to n customers. This is instead of paying for each individual value transfer, more specifically: product transportation to only *one* customer. In short, the Seller pays the Carrier for n transportation services in *one* payment.

5.4 From Possession Flow Model to Process Model

The final step is to create a process model, given the created possession flow model. We create the process model using the Business Process Model and Notation 2.0 (BPMN) [Owen & Raj, 2003]. The main difference between a possession flow model and a process model is that a process model shows additional tasks that are needed to operate a possession flow model. Figure 5 shows the process model that is based on the possession flow model of our running Web shop example in Figure 4. It is important to understand that a process model represents the same networked service as modeled in the value model, trust model, and possession flow model, but then only from another perspective.

We assume that the actors, the value objects, the value transfers and OR/AND elements are already properly identified in the possession flow model. Thus, we adopt these model concepts from the possession flow model to the process model. However, the symbols of the BPMN differs from the *e³value* modeling language. Therefore we have to map the model concepts of the possession flow model to the model concepts of the BPMN. Since there are more than hundred symbols in the BPMN [Dumas et al., 2013], we will only describe the main symbols that are needed to create a process model. We describe these symbols below using our running Web shop example in Figure 5.

Swim lanes. The internal and external actors are represented as pools. These pools could either be a black box pool or a white box pool. A black box pool does not contain internal details, for example the actor Bank is represented as a black box pool. In contrast, a white box pool includes internal details and therefore gives more information about the execution of the process. In Figure 5 are three actors modeled as white box pools, namely (1) the Customer, (2) the Seller and (3) the Carrier. In addition, white box pools can be divided into so called ‘lanes’ to arrange and classify the tasks within the process. For brevity, we do not further divide the internal actors in Figure 5. [Dumas et al., 2013]

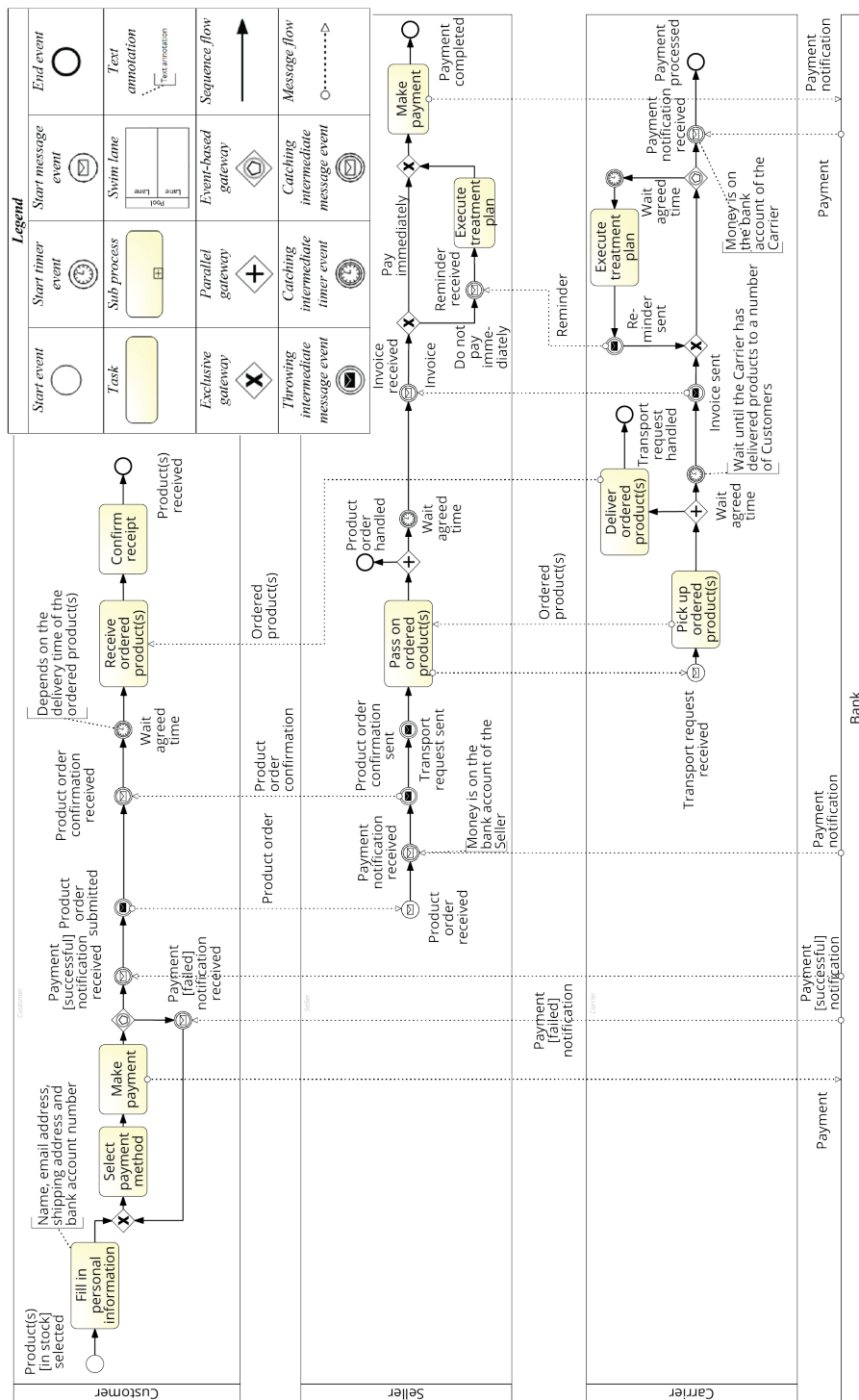


Fig. 5: Process model - Web shop

Activities. Work entities with a duration are represented as tasks in a process model. It is also possible to model tasks that can be expanded, also called sub processes. The value activities, modeled in a possession flow model, can be represented as tasks in a process model. However, it is important to understand that there is a difference between value activities and process model tasks. Value activities are used to model activities that yield profit [Gordijn & Akkermans, 2014], while process model tasks describe an operation. Solely value activities are not sufficient to accomplish the identified value transfers. Besides, not every possession flow model represents value activities, in particular Figure 4. Thus, we need a number of additional tasks to put the possession flow model into operation. In order to model these additional tasks, we answer the following two questions about each identified value activity and/or value transfer: (1) which preceding tasks need to be executed to perform the identified value activity/transfer? (*ex-ante*) and (2) which tasks need to be executed afterwards in order to complete the identified value activity/transfer? (*ex-post*). Considering the Customer white box pool, we see that this actor needs to perform these following tasks in the respective order before (*ex-ante*) making the actual payment to the Seller: (1) select product, (2) fill in personal information and (3) select payment method. To complete (*ex-post*) the payment, the Customer has to receive a payment notification from the Bank that states the payment was successful. [Dumas et al., 2013]

Events. Beside tasks, a process model allows to represent events as well. The difference between tasks and events is that events take place instantly in a process, so there is no duration. An event is either (1) a start event, representing the begin of the process instance (token created) or (2) an intermediate event, indicating an event that happens during a process (token on hold until the event occurs) or (3) an end event, signifying the end of the process instance (token destroyed). Two event examples are message events and timer events. Tasks that sends or receives messages can be replaced by a message event. For example, the Customer submits a product order. This send task is replaced by a *throwing* message event ‘Product order submitted’. The corresponding receive task is depicted as a *catching* message event ‘Product order received’ in the Seller lane. Next, the Customer receives a product order confirmation from the Seller. Then the Customer has to wait until the ordered product is delivered. This temporal interval is represented by a timer event. The process instance can only proceed if this timer event elapses. Every timer event depicts a *catching* event because the timer event elapse is outside the process’ control. [Dumas et al., 2013]

Gateways. The already identified OR-elements and AND-elements in the possession flow model are represented as exclusive gateways and parallel gateways respectively in the process model. Both gateways can either be a split (fork) gateway (arrived tokens diverge) or a join gateway (arrived tokens merge). A split gateway has one incoming branch and more than one outgoing branches. A join gateway is the opposite of the split gateway since it contains more than one incoming branches and one outgoing branch. [Dumas et al., 2013]

Connecting objects. The flow objects, events, tasks and gateways, within the boundary of a pool are connected with the aid of sequence flows. Message flows are used to capture the interaction between the two different black and/or white box pools. For instance, the following two message exchanges between two white box pools are connected via a message flow: ‘Product order confirmation sent’ in the Seller pool and ‘Product order confirmation received’ in the Customer pool. It is important to note that message flows only represent the flow of information between the different pools. [Dumas et al., 2013]

6 Treatment: International Clearing and Distribution of Music Rights

Conform the TAR method, we now apply the treatment as discussed in Sec. 5 to a real-life context. This real-life context is about music Intellectual Property Rights (IPR) clearing and distribution of so-called neighboring rights on music. This is a networked value constellation by nature, as many different actors are involved, and therefore suitable for our purpose. Moreover, there is already detailed knowledge about the existing processes and value models relating to neighboring rights management, which allows us to validate the produced process model by applying our treatment.

Our business partner is an IPR Society who focuses on the right to make music content public. Radio stations, restaurants, shops, etc., collectively referred to as right users, make music public (e.g. they play music at their premises/on the radio) and earn money while doing so (e.g. by advertisements, creating a good shopping atmosphere, etc.) The played music has right owners, for instance artists, producers, and sing & song writers. Right societies act as the intermediate party between right users and right owners: they collect money from music right users, and pay the collected fees to the right owners.

Our business partner, the IPR society, performs two value activities namely (1) clearance of music rights and (2) distribution of collected money to the right owners. In other countries, it is also possible that two different societies perform these two tasks. Clearance implies that the IPR Society collects fee from IPR users, e.g. restaurants and radio stations, on behalf of national and international IPR owners. The IPR Society provides the Right To Make Public (RTMP) to the IPR users in return. Distribution entails that the IPR Society divides the collected fee over the IPR owners, such as artists and producers, based on how many times a music track is played by radio stations and other entities. To obtain play-list data, the IPR Society collaborates with a market research company, radio stations and IPR Sister Societies abroad. [Razavian & Gordijn, 2015]

6.1 IPR Value Model

Razavian and Gordijn (2015) [Razavian & Gordijn, 2015] have studied the IPR e-service and created a value model for handling music rights, using the

e³value modeling language. We extend this value model by adding the IPR Sister Society as an actor to show the international exchanges as well. Figure 6 shows the extended value model for the IPR on music. It is important to understand that this model does not show the time ordering of the value transfers, but only the causality dependency. The IPR Society² transfers value objects with the following seven actors: (1) the Restaurant in the Netherlands, (2) the Radio station, (3) the Market research company, (4) the IPR Sister Society, (5) the Artist, (6) the Producer and (7) the Bank. Note that the Bank is modeled three times in Figure 6 because of pragmatic quality reasons (structured layout).

IPR user - the Restaurant. There are multiple restaurants that play background music in the Netherlands. Therefore, the Restaurant³ actor is modeled as a market segment. The Restaurant has to clear intellectual property rights for the played background music in public. To do so, the Restaurant pays a certain amount of money to the IPR Society. In return the Restaurant receives the RTMP. Note that the IPR Society clearing activity does not transfer value objects (payment service and money) with the Bank. The Restaurant pays money to the IPR Society, thus the Restaurant has to pay the Bank for the payment service. This payment to the Bank by the Restaurant is not modeled in Figure 6 since this value model only captures the value transfers from the perspective of the IPR Society. The music stream for the background music in the Restaurant is obtained by a background music provider, but for the simplicity this actor is omitted in the value model. From a society perspective, a background music provider is treated the same as a Restaurant. Between the clearing and the distribution activities are money and value objects (RTMP) transferred. The clearing activity gives the collected money, obtained from the Restaurant, to the distribution activity (to be discussed later).

IPR Sister Society. In addition, the IPR Society might obtain money from the IPR Sister Society for the international use of music (*see note #4.1*). The IPR Society provides the RTMP on behalf of the Dutch Artists and Producers in return. There are a number of IPR Sister Societies, hence the actor is modeled as a market segment. The IPR Sister Society provides the RTMP on behalf of the IPR owners abroad as well (*see note #4.2*). The IPR Society transfers the collected money from the IPR users in return. This payment is done per transaction via a Bank. Thus, the IPR Society pays a transaction fee to the Bank for the payment service (*see note #4.3*). The IPR Sister Society divides the obtained money over the IPR owners abroad. We assume that the international distribution works the same way as the national distribution, though not modeled.

IPR owners - Artist and Producer. Furthermore, the distribution activity divides the collected money over the IPR owners, in particular the Artist and the Producer (*see note #4.4*). Both actors are modeled as market segments since there are numerous Dutch artists and producers. The money is divided

² ‘IPR Society’ refers to the Intellectual Property Rights Society in the Netherlands.

³ ‘Restaurant’ refers to a Restaurant in the Netherlands.

over m artists (*see note #4.5*) and n producers (*see note #4.6*). We use this construction because usually m (positive integer) artists and n (positive integer) producers are involved in the production of a music track. Note that only the right holders obtain money from the IPR Society. This means that if and only if a music track of a particular IPR owner has been played in public by an IPR user, then the IPR owner obtains money from the IPR Society. For the actual payment, per transaction, the Bank provides a payment service. Besides the payment to the IPR owners, the IPR Society pays the Bank a transaction fee as well (*see note #4.7*).

Radio station and Market research company. In order to properly divide the collected money over the right holders, the IPR Society requires play lists from Radio Stations. The play lists indicate the number of times a music track has been played, thus which national IPR owners, Dutch Artists and Producers, or IPR owners abroad need to be paid. The IPR Society obtains play lists from k (positive integer) important Radio Stations (*see note #4.8*) yearly. As there are multiple radio stations, the Radio station actor is denoted by a market segment. A Radio Station also makes music content public and therefore the Radio Station has a contract with the IPR Society. It is part of the contract that the play list should be delivered by the Radio Station. Moreover, a Market research company provides information about the music usage (*see note #4.9*) once a year. With the aid of the music usage info, the IPR Society gains more insight into the played music tracks by the IPR users (except for the Radio stations). In return the IPR Society pays money to the Market research company via a Bank. To this end, the IPR Society is charged for the use of the payment service by the Bank (*see note #4.10*).

6.2 IPR Trust Model

The next intermediate step in the step-wise method is creating the trust model. In Figure 7 the trust model is shown. Note that, due to lack of space, we only show a fragment of the constructed model. The interested reader is referred to [Hotie, 2015] for the complete model. The involved actors remain unchanged in Figure 7, but the trust model represents the time ordering of the value transfers as well.

The IPR Society transfers value objects with the Radio station and the Market research company simultaneously (*see note #5.1*). The IPR Society pays a transaction fee to the Market research company and the Bank in parallel (*see note #5.2*). In return the Bank provides a payment service and the Market research company sends the music usage info to the IPR Society. Also, the IPR Society signs contracts with k Radio stations (*see note #5.3*) and in return the IPR Society receives play lists.

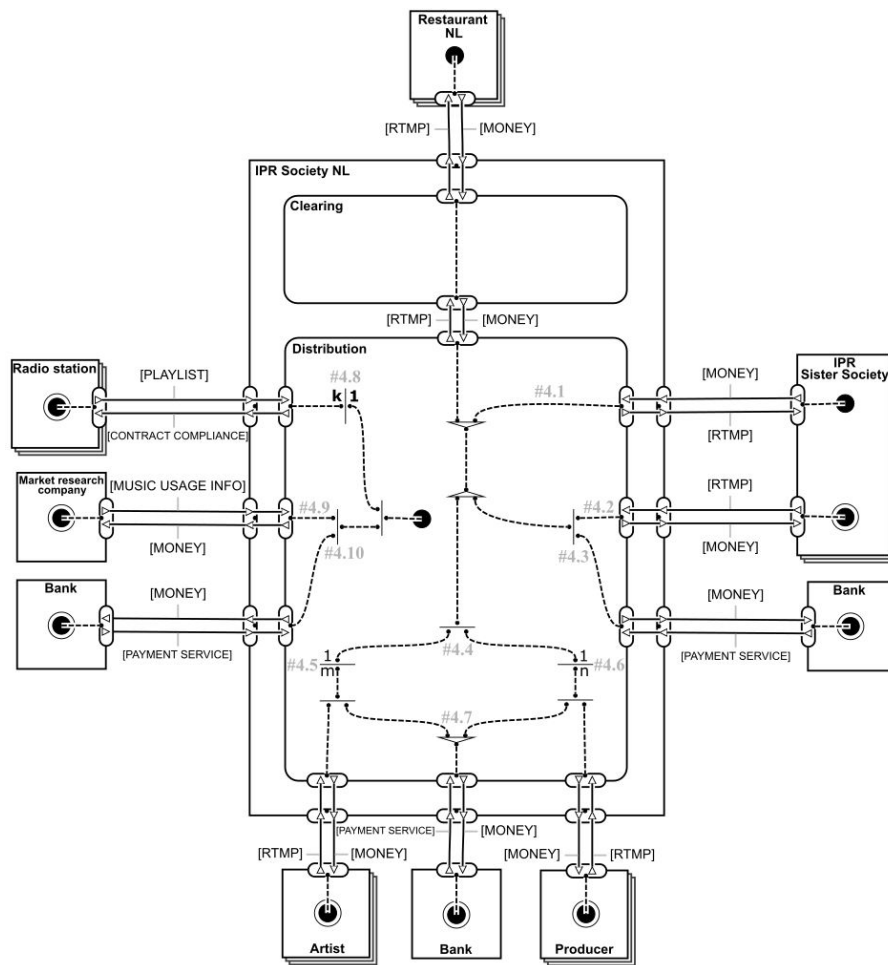


Fig. 6: Value model - IPR on music

6.3 IPR Possession Flow Model

Figure 8 shows a part of the possession flow model. This model represents the physical flow of objects as well. Recall that the physical possession is not the same as ownership. Figure 8 is created using the trust model in Figure 7. Again, the actors remain unchanged. Additionally, the money possession transfer is represented in Figure 8. The payment from the IPR Society to the Market research company is done via a Bank (*see note #6.1*).

Note that the ‘payment service’ value transfer is omitted in Figure 8. For each payment service, the Bank keeps the transaction fee and then sends the remaining agreed amount of money to the respective receiving actor. In other words: the Bank provides a payment service to the IPR Society, even though

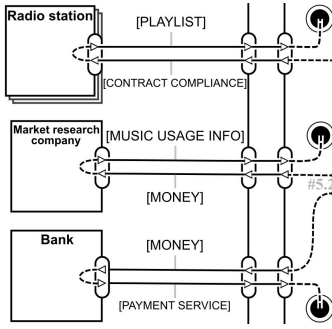


Fig. 7: Trust model - Radio station and Market research company

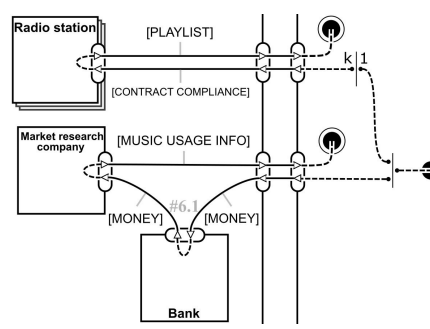


Fig. 8: Possession flow model - Radio station and Market research company

the value transfer ‘payment service’ is not explicitly modeled. The other value objects, value transfers and dependency elements remain the same as in Figure 7. A concluding remark is that aggregations of value transfers are not modeled in Figure 8. For example, payment aggregations are not needed in this specific case since the IPR Society has an agreement with the Bank to pay per transaction, thus not for example once per month.

6.4 IPR Process Model

Given the possession flow model, we create three corresponding process models. We adopt the same actors, value objects, value transfers and OR/AND elements from the possession flow model to the process models. In addition to the possession flow model, the process models include additional tasks as well.

Figure 9 shows the process model that is based on Figure 8. The IPR Society, the Radio Station and the Market research company are depicted as white box pools. Another actor in this model is the Bank, represented as a black box pool. The process starts when the IPR Society needs music usage info and a play list. The process of receiving music usage info from the Market research company and the process of receiving a play list from the Radio station occurs simultaneously.

In order to receive music usage info, the IPR Society has to send a request to the Market research company. Then, the Market research company sends an invoice. The IPR Society either pays immediately or not. The payment occurs via the Bank. The Market research company receives a payment notification from the Bank that states that the money is transferred. If the Market research company has not received a payment notification from the Bank within a month, then a treatment plan is executed and a reminder is sent to the IPR Society. As soon as the Market research company receives a payment notification from the Bank, the Market research company conducts the research on music usage. When the research is completed, the Market research company sends the music usage info to the IPR Society and thereby the process ends.

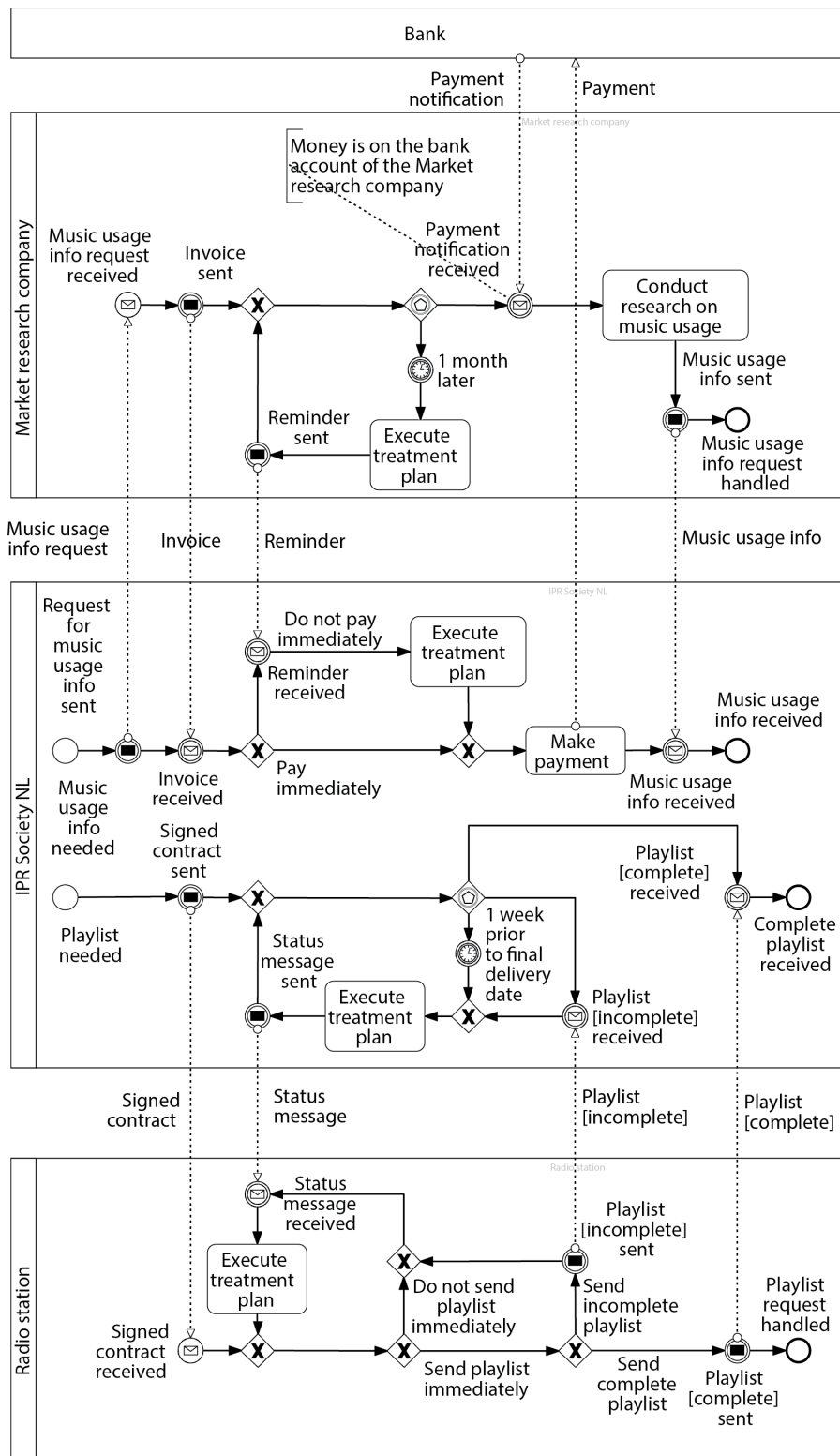


Fig. 9: Process model - Radio station and Market research company

Also, the IPR Society sends the signed contract to the Radio stations. Note that the process of signing the contract is omitted in this model for brevity. After sending the signed contract, the IPR Society either (i) receives a complete play list, or (ii) receives an incomplete play list, or (iii) has not received a complete nor an incomplete play list one week prior to the final delivery date. In the last case (iii), the IPR Society executes a treatment plan and sends a status message to remind the Radio station that nothing has been sent yet. Then again, one of the above mentioned events occur. In case the IPR Society receives an incomplete play list, then the IPR Society sends a status message as well. Finally, the process ends when the IPR Society receives the complete list, possibly after a number of iterations.

We create this process model using the provided information from the IPR employees. In other words: the Market research company and the Radio station white box pools are modeled based on assumptions from the IPR employees. To give more insight into the processes as whole, we choose to model these actors as white box pools anyway. To keep the models to a minimum level of complexity, we depict a number of decisions as exclusive gateways (thus explicit decisions).

7 Treatment Reflection: Evaluation and Improvement

The last step of the TAR method is (1) to evaluate the treatment, and (2) improve the treatment based on this evaluation.

7.1 Treatment Evaluation

For treatment evaluation, we have to understand if the step-wise method to derive a process model based on value model produces an acceptable process model for the IPR society.

In order to ensure that the models provide a truthful description of the online networked services of the IPR Society, other employees of the IPR Society than the employees involved in model construction validated the models. We interviewed three IPR employees who are experts on the respective service parts. This led to a number of adjustments in the models. With the aid of the provided feedback of the one of the employees, we specified two additional timer events in the BPMN model. Timer events are hard to derive from the value model, trust model and possession flow model, as they reflect ‘time outs’, which are not visible in the intermediate models. Another employee validated the models regarding the IPR owners and the IPR Sister Society. This review session led to a number of detailed adjustments concerning the payments between the actors. Finally, the third employee validated the models regarding the IPR users, in particular the Restaurant, during a conference call. After this review session, we specified a timer event label in the IPR user business process model.

This leads to the conclusion that the step-wise method is capable of deriving a reasonable first process model based on value model. However, once this process model is known it is a starting point for a more detailed specification, e.g. concerning handling requests which are not timely responded to, and a more detailed specification of the payment process.

A further comment was that the models are on a fairly global level. For instance reminders and their follow-up actions are not modeled step by step, but are merged to so called ‘treatment plans’. We made a trade-off between simplicity and completeness. Since complete models are often cluttered as well, we decided to choose for simplicity and therefore uncluttered models. Note that this fits well with the philosophy of *e³value* because models that are created to explore a service, are created on a relatively global level.

7.2 Treatment Improvement

Our proposed step-wise method enables to *manually* derive a process model from a value model. Observations regarding the step-wise method, the results, lessons learned and improvements are discussed in the remainder of this section. Based on these observations, we can learn a number of lessons.

Lesson 1: Predetermined decisions

Observation. We observed that certain actors, such as the Bank, have been added at a later stage of the step-wise method. The detail level of particular actors have been changed as well.

Lesson. Before starting to create a model, a number of aspects need to be predetermined and remain unchanged. In this study we learned that it is important to determine in advance which actors are modeled and at what level of detail. This way, it can be avoided to spent extra time on unnecessary things.

Lesson 2: Iterative process

Observation. Minor adjustments were made to the created value model, trust model, possession flow model and the process models until the very last end of the execution of the treatment. For example, we gained new insights during the review sessions with the IPR employees and therefore we adjusted some parts of the models.

Lesson. The iterative process of creating the four model types showed to be an effective method. Iteratively creating and adjusting the models enabled us to constantly compare the value model, the trust model, the possession flow model and the process models. This way, we could ensure that the models are consistent and aligned to each other.

Improvement. The stakeholders mentioned that there are exceptions that have not been taken into account in the models. In order to make the models more complete by detailing further, we could address these exceptions in a subsequent step.

Lesson 3: Processes on a global level

Observation. We deliberately created the process models on a global level because the focus was on the main tasks within the processes. The stakeholders understood the process models and were able to give useful feedback and input.

Lesson. Creating the models on a global level showed to be a convenient and practical method. Only the most essential and relevant tasks should be modeled. The models do not have to be larger than necessary and not every detail has to be included. This makes it easier to understand for the stakeholders.

Improvement. Additional tasks can be added as a sub-process (e.g. a sub-process model for drawing up a contract) to the existing process models. This way, the process models are more complete and yet each model is not infinitely large and complicated.

Lesson 4: Number of model constructs used

Observation. To create the corresponding process model, we used a lot more model constructs in comparison with the amount of model constructs used in the value model, the trust model and the possession flow model. This is intelligible since additional tasks are modeled in a process model. Yet we can conclude that this difference is remarkably big, because we had to create three separate process models to represent all the required tasks for the IPR online networked services. This is despite the fact that we excluded a number of process tasks and simplified certain process parts in the process models.

Lesson. Even though a process model only captures the essentials, it requires more model constructs in comparison with the other previously created models. Thus, the size difference, in terms of used amount of model constructs, between a process model and the other created models is inevitable.

8 Conclusion

In this study, we proposed a step-wise method that enables to *manually* derive a process model from a value model. In this method the e^3 value model is the point of departure. To represent the time ordering of the value transfers, which is not part of the e^3 value methodology, we create a trust model. The next step in the construction of a process model is the physical possession flow of value objects. To this end, we introduce a possession flow model.

We illustrated our step-wise method with the aid of two online networked services of the Intellectual Property Rights society, namely the right clearance and distribution over the IPR owners. The method as well as the constructed models were validated by the IPR society.

The method was tested with one elaborate field study, so further validation is needed. The method can be improved by allowing iterative development, as to our experience, value models and process models are developed side-by-side rather than in a sequential process. Additionally work on model-consistency

checking can be integrated with our method, such that, even in every stage, the formal consistency between value models and process models can be validated.

We believe our method is usable by practitioners, although we did not validate this explicitly. All models are constructed by the researchers, in close cooperation with the IPR society, cf. the TAR method. Extensive validation of the method is a topic of research. However, the *e³value* itself has already a standing tradition and is used in the field. Since the intermediate models for trust and possession are very close to *e³value*, we expect that the method should be usable by practitioners.

Acknowledgments The authors would like to thank the IPR society for the valuable time and insights. Additionally, we are grateful to the constructive remarks of the anonymous reviewers.

References

- [Bodenstaff, 2010] Bodenstaff, L. 2010. Managing Dependency Relations in Inter-Organizational Models. PhD thesis, University of Twente.
- [Bodenstaff et al., 2007] Bodenstaff, L., A. Wombacher, & M.U. Reichert 2007. On Formal Consistency between Value and Coordination Models.
- [Buhr, 1998] Buhr, Raymond JA 1998. Use case maps as architectural entities for complex systems. *Software Engineering, IEEE Transactions on*, 24(12):1131–1155.
- [Dumas et al., 2013] Dumas, Marlon, Marcello La Rosa, Jan Mendling, & Hajo A Reijers 2013. *Fundamentals of business process management*. Springer.
- [Geerts & McCarthy, 1999] Geerts, Guido L., & William E. McCarthy 1999. An accounting object infrastructure for knowledge-based enterprise models. *IEEE Intelligent Systems*, pages 1–6.
- [Gordijn & Akkermans, 2001] Gordijn, Jaap, & Hans Akkermans 2001. Designing and evaluating e-business models. *IEEE intelligent Systems*, 16(4):11–17.
- [Gordijn & Akkermans, 2003] Gordijn, Jaap, & Hans Akkermans 2003. Value based requirements engineering: Exploring innovative e-commerce idea. *Requirements Engineering Journal*, 8(2):114–134.
- [Gordijn & Akkermans, 2014] Gordijn, Jaap, & Hans Akkermans 2014. *Value Webs, Understanding e-Business Innovation*. self published.
- [Hevner et al., 2004] Hevner, Alan R., Salvatore T. March, Jinsoo Park, & Sudha Ram 2004. *Design Science in Information Systems Research*. *MIS Q.*, 28(1):75–105.
- [Hotie, 2015] Hotie, Felicia 2015. *From Business Value Models to Business Process Models*. Master’s thesis, Vrije Universiteit Amsterdam.
- [Huemer et al., 2011] Huemer, Christian, et al. 2011. UML Profile for UN/CEFACT’s Modeling Methodology (UMM) Foundation Module Version 2.0 Technical Specification 2011-04-01. UNECE, http://www1.unece.org/cefact/platform/download/attachments/44204303/Specification_UMM_Foundation_Module_V2.0_TechnicalSpecification.pdf.
- [Ionita et al., 2015] Ionita, Dan, Roel J Wieringa, Lars Wolos, Jaap Gordijn, & Wolter Pieters 2015. Using value models for business risk analysis in e-service networks. In *The Practice of Enterprise Modeling*, pages 239–253. Springer International Publishing.
- [Kartseva et al., 2009] Kartseva, Vera, Jaap Gordijn, & Yao-Hua Tan 2009. Designing Value-based Inter-organizational Controls Using Patterns, pages 276–301. Volume 14 of [Lyytinen et al., 2009].
- [Lyytinen et al., 2009] Lyytinen, Kalle, Peri Loucopoulos, John Mylopoulos, & Bill Robinson (eds) 2009. *Design Requirements Engineering: A Ten-Year Perspective*. LNBIP. Springer Verlag.
- [Mendoza & Pracejus, 1997] Mendoza, Norma A, & John W Pracejus 1997. Buy now, pay later: Does a future temporal orientation affect credit overuse? *Advances in Consumer Research*, 24:499–503.

- [Meyer, 1985] Meyer, B. 1985. On Formalism in Specifications. *IEEE Software*, 2(1):6–26.
- [Mohan & Ramesh, 2003] Mohan, Kannan, & Balasubramaniam Ramesh 2003. Ontology-based Support for Variability Management in Product and Service Families. In *Proceedings of the 36th Hawaii International Conference on System Sciences*, Hawaii.
- [Mylopoulos et al., 1990] Mylopoulos, John, Alexander Borgida, Matthias Jarke, & Manolis Koubarakis 1990. Telos: Representing Knowledge About Information Systems. *ACM Trans. Inf. Syst.*, 8(4):325–362.
- [Normann & Ramírez, 1993] Normann, R., & R. Ramírez 1993. From Value Chain to Value Constellation: Designing Interactive Strategy. *Harvard Business Review*, (july-august):65–77.
- [Normann & Ramírez, 1994] Normann, R., & R. Ramírez 1994. *Designing Interactive Strategy - From Value Chain to Value Constellation*. John Wiley & Sons Inc., Chichester, UK.
- [OMG, 2011] OMG 2011. Business Process Model and Notation (BPMN). <http://www.omg.org/spec/BPMN/2.0/PDF/>. [Online; accessed 17-January-2017].
- [Osterwalder & Pigneur, 2010] Osterwalder, Alexander, & Yves Pigneur 2010. *Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers*. Alexander Osterwalder & Yves Pigneur.
- [Owen & Raj, 2003] Owen, Martin, & Jog Raj 2003. BPMN and business process management. *Introduction to the New Business Process Modeling Standard*.
- [Pijpers & Gordijn, 2007a] Pijpers, Vincent, & Jaap Gordijn 2007a. Bridging Business Value Models and Business Process Models in Aviation Value Webs via Possession Rights. In *Proceedings of the 20th Annual Hawaii International Conference on System Sciences*, page cdrom. Computer Society Press.
- [Pijpers & Gordijn, 2007b] Pijpers, Vincent, & Jaap Gordijn 2007b. Bridging business value models and process models in aviation value webs via possession rights. In *System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on*, pages 175a–175a. IEEE.
- [Pijpers & Gordijn, 2008a] Pijpers, Vincent, & Jaap Gordijn 2008a. Consistency Checking Between Value Models and Process Models: A Best-of-Breed Approach. In Johanneson, Paul, Jaap Gordijn, Ela Hunt, Xavier French, & Remi Coletta (eds), *Proceedings of the Third International Workshop on Business/IT Alignment and Interoperability (BUSITAL'08) held in conjunction with CAiSE'08 Conference*, pages 58–72. CEUR-WS.org.
- [Pijpers & Gordijn, 2008b] Pijpers, Vincent, & Jaap Gordijn 2008b. Consistency checking between value models and process models; a best-of-breed approach. *Proceedings of BUSITAL*, 8:59.
- [Razavian & Gordijn, 2015] Razavian, Maryam, & Jaap Gordijn 2015. Consonance Between Economic and IT Services: Finding the Balance Between Conflicting Requirements. In *Requirements Engineering: Foundation for Software Quality*, pages 148–163. Springer.
- [Schuster & Motal, 2009] Schuster, Rainer, & Thomas Motal 2009. From e3-value to REA: Modeling Multi-party E-business Collaborations. In *Proceedings of the 2009 IEEE Conference on Commerce and Enterprise Computing, CEC '09*, pages 202–208, Washington, DC, USA. IEEE Computer Society.
- [Schuster et al., 2010] Schuster, Rainer, Thomas Motal, Christian Huemer, & Hannes Werthner 2010. From Economic Drivers to B2B Process Models: A Mapping from REA to UMM. In Abramowicz, Witold, & Robert Tolksdorf (eds), *Business Information Systems, 13th International Conference, BIS 2010, Berlin, Germany, May 3-5, 2010. Proceedings*, volume 47 of *Lecture Notes in Business Information Processing*, pages 119–131. Springer.
- [Snijders & Rank-Berenschot, 2001] Snijders, H.J., & E.B. Rank-Berenschot 2001. *Goederechtigd. Kluwer*, Deventer.
- [Weigand et al., 2006] Weigand, H., P. Johannesson, B. Andersson, M. Bergholtz, A. Edirisuriya, & T. Ilayperuma 2006. On the notion of value object. In Dubois, E., & K. Pohl (eds), *Advanced Information Systems Engineering. Proceedings of CAiSE 2006*, LNCS, 4001, pages 321–335, Heidelberg, DE. Springer-Verlag.
- [Weigand et al., 2007a] Weigand, H., P. Johannesson, B. Andersson, M. Bergholtz, A. Edirisuriya, & T. Ilayperuma 2007a. Value modeling and the transformation from value model to process mode. In Doumeingts, G., J. Mller, G. Morel, & B. Vallespir (eds), *Enterprise Interoperability: New Challenges and Approaches*, pages 1–10, London, UK. Springer-Verlag.

-
- [Weigand et al., 2007b] Weigand, Hans, Paul Johannesson, Birger Andersson, Maria Bergholtz, Ananda Edirisuriya, & Tharaka Ilayperuma 2007b. Value object analysis and the transformation from value model to process model. In Doumeingts, Guy, Jrg Mller, Grard Morel, & Bruno Vallespir (eds), *Enterprise Interoperability*, pages 55–65. Springer London.
- [Wieggers, 1999] Wieggers, Karl E. 1999. *Software Requirements*. Microsoft Press, Redmond, WA.
- [Wieringa & Gordijn, 2005] Wieringa, Roel, & Jaap Gordijn 2005. Value-oriented design of correct service coordination protocols. In *Proceedings of the 20th ACM Symposium on Applied Computing*, pages 1320–1327. ACM Press.
- [Wieringa, 2014] Wieringa, R. J. 2014. *Design science methodology for information systems and software engineering*. Springer Verlag, London.