A UML Profile for the e3-Value e-Business Model Ontology

C. Huemer[1], A. Schmidt[2], H. Werthner[1], and M. Zapletal[1]

[1] Institute of Software Technology and Interactive Systems, Vienna University of Technology, Austria
huemer@big.tuwien.ac.at, werthner@ec.tuwien.ac.at, marco@ec.tuwien.ac.at
[2] SAP Research CEC, St. Gallen, Switzerland
al.schmidt@sap.com

Abstract. Shorter life cycles of products and services require faster changing business models. Information systems must quickly adjust to the adapted business models. Business models are usually described by their own proprietary notation, which is incompatible with UML - the de-facto modeling standard in software engineering. In order to allow a straight-through modeling approach from business models over business process models to software artifacts, it is desirable to use a common modeling approach. Thus, we suggest to map existing concepts to describe business models onto the UML notation. In our work we mainly focus on inter-organizational systems. A promising approach describing a business model for an inter-organizational network of actors is delivered by e3-Value. In this paper, we present a discussion of different approaches to represent the e3-Value concepts by means of UML. A UML notation for e3-Value is a precondition to future work on aligning e3-Value to UML-based approaches specifying inter-organizational business processes.

Key words: Business modeling, e3-Value , UML profile, B2B

1 Motivation

Service-oriented Architectures (SOA) are said to provide a means of quickly adapting IT to changing business needs. However, most approaches focus only on the IT layer. Instead, we propose an integrated approach for inter-organizational systems spanning from business models over business process models to their execution in a SOA. An integrated approach starts with analyzing the economic drivers for a business collaboration. This means describing the business models in terms of economic values that are exchanged between business partners. Candidate approaches to be used on this level of abstraction are e3-Value [1], REA [2] or BMO [3].

In order to guarantee that each partner deserves its economic value they have to agree with each other on the inter-organizational business processes to realize the value exchanges. A resulting global choreography may be described by the UN/CEFACT modeling methodology (UMM) which is specifically designed for

this purpose [4]. The UMM is defined as a UML profile. In a following step the orchestration of the internal business processes is defined and bound to the UMM choreography. This orchestration may also be described by the means of UML leading to a software development process that ends up with the automatic generation of machine-interpretable workflows.

In order to allow a straight-through modeling approach, it is desirable to base the different steps in developing inter-organizational systems on a single modeling paradigm. Most of the steps described above are already based on UML. This means they customize the general purpose language UML by means of stereotypes, tagged values and constraints for their specific purpose. Only the definition of the value exchanges is not based on UML. Thus, the definition of the UML profile for value exchanges completes an overall UML based approach for inter-organizational systems. Since the e3-Value approach specifically targets business models in an inter-organizational environment, it is our goal to transform its concepts to a UML profile. In this paper we discuss different options for representing e3-Value in UML. A UML-based e3-Value notation is a precondition to seamlessly integrate it with the UMM. However, a detailed analysis of this integration is out of scope for this paper and is up to future work.

2 e3-value at a glance

2.1 Basic concepts

e3-Value is an ontology-based methodology for modeling and designing business models for business networks [1] incorporating concepts from requirements engineering and conceptual modeling (including a graphical notation). Its main focus is on identifying and analyzing how value is created, exchanged and consumed within a multi-actor network, hence, taking the economic value perspective and visualizing what is exchanged (which kind of economic value) by whom [5]. An economic value exchange, and consequently the e3-Value ontology as a whole, is based on the principle of reciprocity emphasizing the dual character of business transactions. This "give and take"-approach denotes that every actor offers something of economic value, such as money, physical goods, services, or capabilities, and gets something of economic value in return.

The e3-Value ontology defines a number of concepts that will be briefly outlined in this section. The concepts are described in more detail in [1] as well as in [6] and [5].

Actors represent parties engaged in a value exchange. They are considered as independent economic entities that strive for profitability (in case of an enterprise) or maximizing their economic utility (in case of an end-consumer) by carrying out *value activities*. These profitable or utility-increasing *value activities* are intended to be directly and unambiguously assigned to the corresponding actors. By conducting *value activities* actors exchange *value objects* that are valuable to one or more actors of the business network. As mentioned before, these objects are "things of value" - either material, such as physical goods, products or money, or non-material (e.g. services, capabilities or experience).

Actors signal their will to provide or request *value objects* through interfaces. In the e3-Value ontology these interfaces are called *value ports*. The rationale of *value ports* is to abstract from an actor's internal processes and instead concentrate exclusively on the external connection to other business partners (i.e. actors) and other components. Two *value ports* are connected to each other via a *value exchange*. The latter depicts one or more potential trades of *value objects* between *value ports*. The values offered to or requested from the environment are represented by so-called *value offerings*, representing sets of equally directed *value ports*. The concept of *value offerings* allows the mapping of "object bundling" in case that objects are of value and can be requested or provided only in combination. This means that an offering may consist of several *value objects* to be exchanged. One or a maximum two *value offerings* - in general one incoming and one outgoing - are subsumed by a *value interface* typifying the concept of economic reciprocity and showing which *value object* is offered in return for another. Each *actor* may have multiple *value interfaces* grouping individual *value ports*. The exchange of values is atomic - i.e., *value objects* in an offering are exchanged through the *value interface* on all of its *value ports* (each exchanging exactly one *value object*) or none of them.

For creating appropriate visual representations of the value models a graphical notation is provided - the stated elements are represented in figure 1.

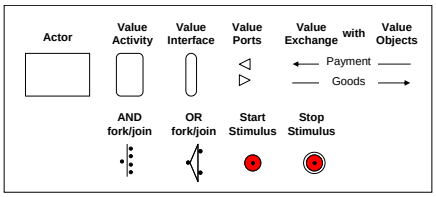

Fig. 1. e3-Value Concepts

For mapping more complex, multi-step scenarios, components of existing scenario techniques, so-called *use case maps* (UCMs), are deployed [7]. These UCMs add four further modeling concepts: Firstly, a *scenario path* indicates via which *value interfaces* objects are exchanged. Each *scenario path* is subdivided into one or more segments. Individual segments are related to each other by *connection elements*. Similarly to well-known process modeling concepts, *AND forks* as well as *OR forks* (and their corresponding *joins*) can be used to model two or more sub-paths. Furthermore, each *scenario path* starts with a *start stimulus*, representing a specific consumer need, and ends with *stop stimulus* after the last segment of the *scenario path*.

2.2 Example: waste management

In this subsection we demonstrate the e3-Value concepts by means of an example from the waste management domain. We consider the international - i.e. cross-border - trading of waste. In this case study waste is traded between an `exporter` and an `importer`. In principle we must distinguish two different kinds of trading. In the majority of cases the `exporter` has to pay the `importer` for the `waste handling`, i.e. for the recovery or disposal of the waste. Accordingly, `waste` and `money` is given to the `importer` and the `exporter` gets the service of `waste handling` in return. In the minority of cases, the waste is traded like a regular good. The trading of re-cycled paper is a typical example of such a case. This means the importer has to pay for the waste. In other words the `importer` gets the `waste`, and the `exporter` receives `money` in return.

Moreover, the international trading of waste has some legal implications. Competent authorities in the countries of the exporter and the importer control the trading. Accordingly, the `exporter` has to inform the `export authority` about a waste transport. The `exporter` delivers relevant `environmental information` about the transport and the `export authority` issues a `transport allowance` in return. The value of a `transport allowance` is considered equally to fulfill the legal regulations and, thus, to avoid possible fines. Similarly, the `importer` has to provide `environmental information` about the transport to the `import authority` in order to get the `transport allowance` for importing the waste. The `export authority` and the `import authority` have a natural interest in providing each other with the relevant `environmental information` collected on each side. Thereby, both competent authorities are able to obtain the full information about the waste transport on each side.

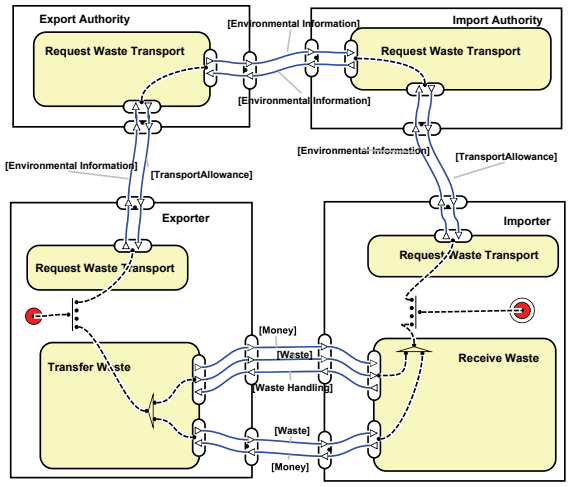

Fig. 2. Waste management example modeled with e3-Value notation

Figure 2 shows the resulting e3-Value diagram for the waste management scenario. Each party - `exporter`, `export authority`, `import authority`, and `importer` - is represented by an *actor* in e3-Value and performs a *value activity* named `request waste transport`. The *actors* conduct *value exchanges* between their `request waste transport` *value activities* in order to fulfill the legal implications. The `exporter` provides `environmental information` to the `export authority` in order to get a `transport allowance` in return. The outgoing *value port* of the exporter's activity indicates that he provides the `environmental information`. Similarly, the incoming *value port* shows that the `exporter` demands a `transport allowance` in return. The atomicity of the *value exchange* is denoted by the *value interface* sitting on the edge of the exporter's `request waste transport` activity. It binds the two *value ports* together - indicating that either both or none of the two *value exchanges* take place. The *value exchanges* between the `export authority` and the `import authority` as well as between the `importer` and the `import authority` - as described above - are modeled in a similar way. `Exporter` and `importer` perform additional *value activities* that represent the actual trading of the waste. The *value activity* `transfer waste` defines two *value interfaces*. The first one provides `money` and `waste` and consumes the service of `waste handling` in return. The second *value interface* trades `waste` against `money`. The *value activity* `receive waste` of the `importer` provides the complementary *value interfaces*.

Furthermore, the concept of *scenario paths* - as depicted in figure 2 - shows the path by which interfaces values are exchanged. If `waste` is traded, the exchange of `environmental information` and `transport allowances` is mandatory. In contrary, there is no information exchange with the competent authorities required if no waste is going to be traded. It follows, that these two *value exchanges* are interlinked by an AND connector. This is denoted by the *AND fork* following the *start stimulus* located in the area of the `exporter`. The first path of the *AND fork* connects the *value interfaces* of the `request waste transport` activities - indicating that all of these *value exchanges* are required in the scenario. The second path starting from the *AND fork* represents the trading of the waste itself. We already outlined the two alternatives: either `waste` in return for `money` or `waste` and `money` in return for `waste handling`. These two alternatives are interlinked by an XOR connector. This is denoted by the *OR fork* preceding the two *value interfaces* of `transfer waste`. Note, an OR fork in e3-Value has an XOR semantic. The scenario paths are merged by an *OR* and *AND join* on the right hand side of figure 2. This means they are merged before the overall scenario is ended with a *stop stimulus*.

3 Mapping e3-Value to UML

This section focuses on mapping the e3-Value concepts to a UML profile. In addition to a prose description of the concepts, e3-Value comes with a MOF-like meta model specifying the concepts and the relations between them (c.f. [5]). Furthermore, e3-Value defines its own graphical notation. The MOF-like meta

model is significantly different from the UML meta model. Developing a UML profile means to represent the e3 concepts on top of the UML meta model by means of stereotypes, tagged values and constraints.

A mapping to a UML profile is not straightforward due to the significant differences between the e3-Value meta model and the UML meta model. Since UML originates from software engineering, none of the UML standard diagrams has originally been intended to capture e3-Value semantics. It is necessary to analyze which of the existing UML standard diagrams and corresponding model elements are best suited for a customization towards UML. In the following subsections we discuss five different alternatives by means of the waste management example and state their strengths and shortcomings.

3.1 The activity parameter variant

Value activities are a cornerstone of e3-Value. At a first glance it seems to be consequent to map them to activities and activity diagrams, respectively. A possible solution for the waste management scenario based on activity diagrams is depicted in Figure 3.

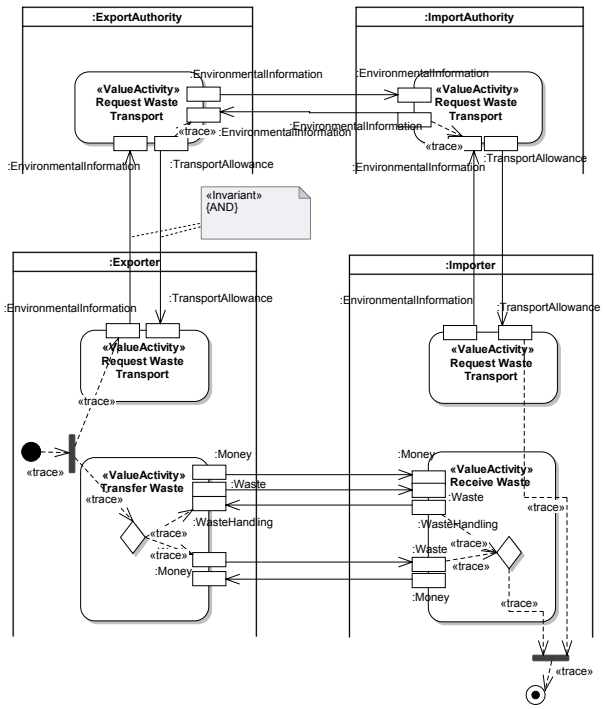

Fig. 3. The activity parameter variant

Each e3-Value *actor* results in a UML activity partition assigned to a corresponding UML actor. The activity partition shows his area of responsibility. In the waste management example the activity diagram includes four activity partitions for the following actors: `exporter`, `export authority`, `import authority`, and `importer`. Each *value activity* is mapped to a UML activity showing the stereotype *value activity*. In the waste management example, each party performs an activity `request waste transport`. Furthermore, the `exporter` performs `transfer waste` and the `importer` performs `receive waste`.

We use UML activity parameters to model *value exchanges*. An activity parameter describes the input or output to/from a UML activity. It follows, that an offering activity carries an output parameter, whereas a consuming activity carries an input activity. The flow of a *value object* is described by an exchange from the output parameter to the input parameter. The *value object* itself is a stereotype based on the UML metaclass class. This *value object* is assigned to both, to the input parameter and to the output parameter. To illustrate these concepts we take a look at the *value exchanges* between the `exporter` and the `export authority` on the left hand side of figure 3. The *value exchanges* are realized between the activities called `request waste transport` on each partner's side. The *value object* `environmental information` is assigned to the output parameter of the `exporter's activity` as well as to the input parameter of the `export authority's` activity in order to realize the *value exchange* from the `exporter` to the `export authority`. The flow of the *value object* `transport allowance` goes the other way round.

A major drawback of this variant is the fact, that it lacks the concepts of *value ports* and *value interfaces*. These concepts are required to group *value exchanges* for denoting the atomicity of a set of *value exchanges*. There is no concept in UML activity diagrams that corresponds to an e3-Value *value port*. For representing *value interfaces* one might think of UML parameter sets to group multiple parameters. However, the semantics of a parameter set allow to group either input or output parameters, but does not allow a mix of input and output parameters. Furthermore, multiple parameter sets of the same activity are in an XOR relationship with each other - which does not correspond to the e3-Value semantics. Due to these two reasons, UML parameter sets do not fit the requirements of our mapping. A workaround denoting the atomicity of two or more *value exchanges* is attaching a constraint to the respective object flows. In figure 3 we defined such a constraint - mandating an AND relationship between the *value exchanges* of the `exporter` and the `export authority`. For the sake of readability we refrain from showing this kind of constraint between other *value exchanges* in figure 3.

For mapping e3-Value *scenario paths* we utilize the pseudo states for modeling flows in UML activity diagrams. The *start stimulus* and the *stop stimulus* of e3-Value are mapped to UML initial and final states, respectively. The *AND fork/joins* are mapped to the UML concepts of fork/joins, whereby e3-Value *OR fork/joins* correspond to UML decisions/merges in our mapping. In figure 3, the decision node within the `transfer waste` activity of the `exporter` indicates

a choice of two different scenarios: either exchanging `waste` for `money` or exchanging `waste` and `money` for the service of `waste handling`. Similarly, the fork node immediately after the initial state denotes that the `exporter` must conduct a *value exchange* with the `export authority` (`environmental information` against `transport allowance`) and a *value exchange* with the `importer` as explained above.

It is important to note that a *scenario path* does not specify a control flow amongst the activities nor does it mandate any sequences in time. In order to stress this fact we refrain from using the UML connector type control flow. Instead, we use the concept of a *trace* - a stereotyped UML dependency - to describe the `scenario paths`. Usually a *trace* should lead to a *value interface*. Since the concept of a *value interface* cannot be represented in this variant a *trace* leads to any *value port* being part of a set of *value ports* that are graphically grouped to a *value interface*.

3.2 The boundaries variant

The boundaries variant is somewhat similar to the activity parameter variant. It is also based on UML activity diagrams. The representation of e3-Value *actors* and e3-Value *value activities* still remains the same. However, as shown in figure 4 *value exchanges* are modeled using the UML object node notation instead of the activity parameter notation. In other words, the exchange of a *value object* is modeled by an object flow starting from the offering activity to the *value object* modeled as object node leading to the consuming activity. Although the seman-

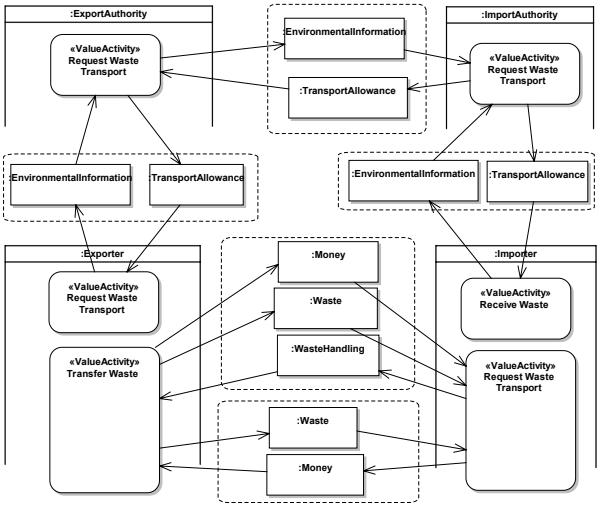

Fig. 4. The boundaries variant

tics is the same as in the activity parameter variant, the object node notation

allows grouping the exchanged *value objects*. We use the concept of interruptible regions provided by UML for describing the atomicity of *value exchanges* being part of a value scenario - e.g., the exchange of `transport allowance` in return to `environmental information` between the `exporter` and the `export authority`. In activity diagrams, interruptible regions usually denote boundaries for exception handling. Nevertheless, we are able to use boundaries to express the semantics of economic reciprocity of two or more *value exchanges*.

The usage of interruptible regions results in a big drawback. Since interruptible regions must not be the source or the target of any UML connector, we are not able to represent *scenario paths*. Similar to the activity parameter variant, the inability to represent the e3-Value concepts of *value ports* and *value interfaces* is a major shortcoming. Although the boundary variant benefits from compact and simple diagrams, it is inadequate for describing more complex e3-Value scenarios.

3.3 The interface variant

The interface variant is also based on UML activity diagrams, but provides another solution for *value exchanges*. As shown in figure 5 this solution makes use of UML interfaces. A *value interface* is a stereotyped UML interface that is now used to bind *value exchanges* to an atomic unit. A *value activity* may have one to many *value interfaces* - each connected with a UML dependency. According to our example in figure 5, the `exporter` has a total of three *value interfaces*. One is bound to `request waste transport`, the remaining two are bound to the *value activity* `transfer waste`. A *value exchange* between two *value interfaces* is described by the UML concept of an information flow. Information flows describe circulation of information in a system in a general manner [8]. In our mapping, they are used to exchange *value objects*. The *value object* being exchanged is specified as the information flow's classifier. As shown in figure 5, the *value interfaces* of `exporter` and `export authority` exchange values via two information flows. One information flow sends `environmental information` from the `exporter` to the `export authority` in exchange for another information flow carrying the `transport allowance`. All classifiers of information flows are stereotyped as *value objects*.

The interface variant also provides *scenario paths* - described by similar concepts as introduced for the activity parameter variant. Although the definition of this variant lacks the explicit notion of *value ports*, it is the first mapping variant described so far that allows for modeling implicitly all e3-Value concepts by means of UML. In terms of *value ports*, UML purists might correctly argue, that the concept of a connector end as defined in the UML meta model corresponds to a *value port* in e3-Value. However, for the sake of simplicity - and since connector ends are not supported by most UML tools - we refrain from stereotyping the connector ends of an information flow as a *value port*.

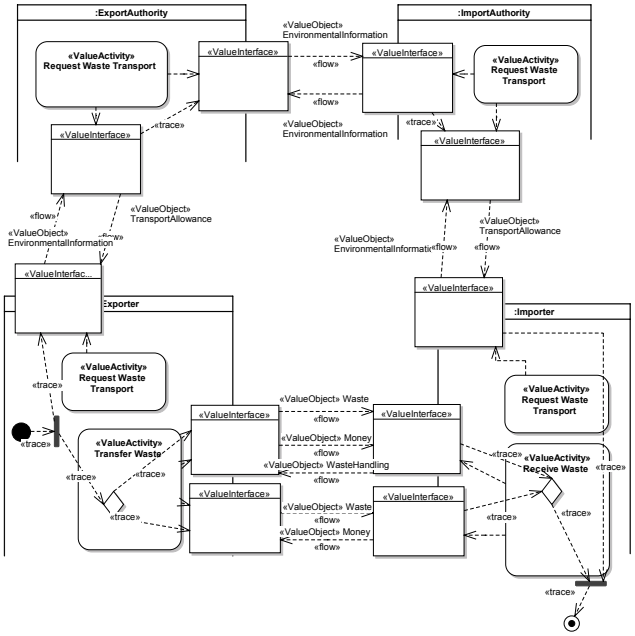

Fig. 5. The interface variant

3.4 The signal variant

Similar to the interface variant, the signal variant (c.f. figure 6) covers all e3-Value concepts. However, it differs from the interface variant in terms of modeling *value interfaces* and *value ports*. In the signal variant, *value interfaces* are represented as stereotypes based on UML activities. *Value interfaces* are modeled as child elements of the *value activity* they belong to.

As shown in our waste management example in figure 6, we place the *value interfaces* inside of the *value activities*. Unlike in the interface variant, *value ports* are modeled explicitly using the concept of UML signals. We use send signal actions to indicate outgoing *value ports* and receive signal actions for representing incoming *value ports*. Each signal action - no matter if incoming or outgoing - is stereotyped as a *value port*. As we know from e3-Value , *value ports* must be part of exactly one *value interface*. Correspondingly, we place the stereotyped UML signals within the *value interfaces* they are part of.

The concepts for representing *value exchanges* and *scenario paths* look alike the ones used in the interface variant. *Value exchanges* are described using information flows. The *value object* that is conveyed through the *value exchange* is assigned as the information flow's classifier. For defining the *scenario path* we apply the concept of the *trace* dependency combined with some elements borrowed from UML diagrams in order to describe *AND* and *OR fork/joins*.

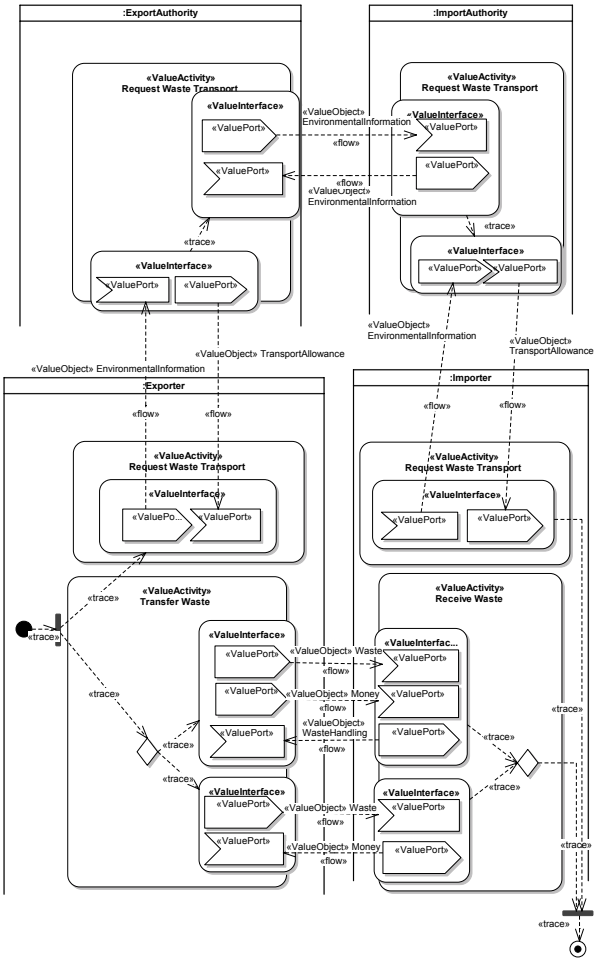

Fig. 6. The signal variant

The signal variant provides the user with all e3-Value concepts when using UML for business modeling. Some UML purists may disagree with the nested UML activities and actions in order to model *value ports* and *value interfaces* of a *value activity*. Those nested activities - missing start and end nodes - do not result in a well defined sequence of activity flows. However, this is a result of the different semantics of flows in e3-Value and in UML activity diagrams.

3.5 The use case variant

In order to overcome the incompliance in the flow semantics between e3-Value and UML activity diagrams, we propose another solution based on UML use case diagrams (c.f. figure 7). An e3-Value *actor* is again represented by a UML actor.

An actor may be connected to one or more use cases representing *value activities*.

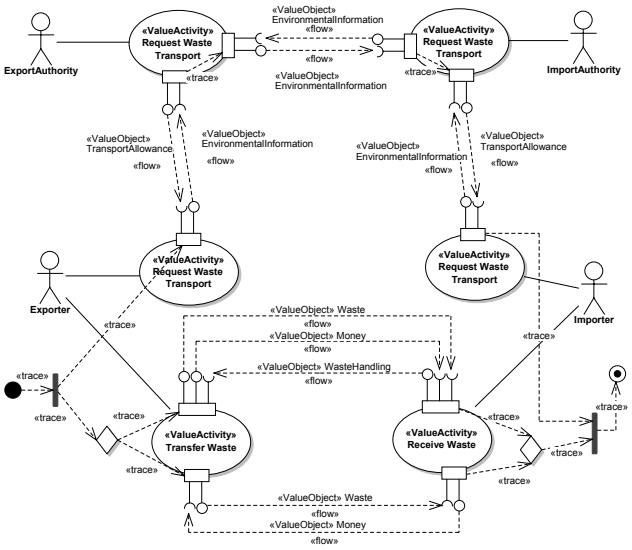

Fig. 7. The use case variant

Thus, each use case gets the stereotype *value activity* assigned. Considering our example in figure 7, we model four *business partners* named `exporter`, `export authority`, `import authority` and `importer`. Each *business partner* is connected to his stereotyped *value activities*. For example, the `exporter` is associated with two use cases - `request waste transport` and `transfer waste` - both stereotyped as *value activity*.

In order to represent the *value interfaces* of a *value activity* we use UML ports. According to the UML2 specification, ports represent interaction points between a classifier and its environment [8]. Since the concept of a use case inherits from the concept of a classifier, a use case is eligible to have embedded ports. Each port in our mapping is stereotyped as *value interface*. In UML, each port may define zero to many interfaces. Each interface is either providing or requiring objects. Thus, the concept of a UML port interface perfectly fits the needs of an e3-Value *value port*. Accordingly, outgoing ports become providing interfaces and incoming ports become consuming interfaces. Each port interface gets the stereotype *value port* assigned. Due to readability reasons, we do not show the stereotypes *value interfaces* and *value ports* in our example figure 7.

Consider the *value activity* `transfer waste` of our example: This use case has two ports representing its *value interfaces*. One port shows the exchange of `waste` and `money` in return to `waste handling`. Thus, this port defines three port interfaces - two providing ones indicating that `waste` and `money` is transfered

to the *value activity* `receive waste` and one interfaces consuming the service of `waste handling`. The other port of `transfer waste` has two interfaces - one provides `waste` and the other one consumes `money` in return.

The *value exchanges* between the ports and their interfaces are described like in the interface and signal variant. We denote them by information flows leading from providing interfaces to consuming interfaces. The classifier of this information flow corresponds to the *value object* sent in this *value exchange*. *Scenario paths* are again described by the concept of a *trace* dependency. We use *trace* dependencies to describe possible paths of *value interfaces* with UML pseudo states representing *AND/OR fork* and *joins*.

In summary, we prefer the use case variant for mapping e3-Value models to UML. It results in comprehensible business models representing the whole set of e3-Value semantics. In addition, the use case variant is fully compliant to the UML meta model. Furthermore, e3-Value is a method for requirements gathering and will be used as such as part of the UMM. In the UML - and also in the UMM - the concept of a use case serves as a mean for eliciting the requirements of a system. Thus, it is reasonable to model e3-Value concepts by means of use case diagrams rather than by activity diagrams.

4 Related Work

There have been a lot of different approaches to model business by means of UML. However, most of the processes focus on modeling business processes. A good evaluation of different approaches of modeling business processes by means of UML activity diagrams is provided in [9]. However, there is a significant difference between business modeling and business process modeling [10]. The former describes the value perspective, whereas the latter specifies the process perspective. Hence, our approach may sit on top of the presented business process modeling approaches. The only all-embracing UML-based approach to model businesses and their processes is provided by Penker and Eriksson [11]. They show how to use UML for documenting the entire enterprise. It is outlined how to model businesses, from business architecture to processes, business rules, and goals. Furthermore they define business patterns that provide re-usable solutions to common business problems.

In the research field of ontology-based e-business modeling other non-UML-based concepts, similar to e3-Value, have been proposed in the past. The REA (Resource-Event-Agent) business model ontology evolved from a generalized framework for modeling accounting information systems [12] to an ontology for enterprise information systems [13]. Their main components - agents, resources, events and exchanges - are essentially equivalent to the corresponding e3-Value concepts [5]. The (e-)Business Model Ontology (BMO), in turn, possesses a much wider scope conceptualizing a variety of internal resources, assets and capabilities [3] [14]. Finally, in [15] the authors introduce a reference ontology by incorporating concepts from e3-Value, REA and BMO.

The identification of the objects of value exchanged within a business network is not always straightforward. In our example, participants exchange documents fulfilling legal regulation and avoiding fines. These documents are objects of value. A discussion about the notion of value objects is given in [16].

Further ontological approaches comprise the AIAI Enterprise Ontology [17], which defines business model-related concepts, including activities and sales (in accordance to e3-Value), but is much more complex with a large number of concepts and relationships. Likewise, the Toronto Virtual Enterprise (TOVE) ontology covers a wide area of company-related concepts [18]. Still, important cross-organizational aspects (e.g. an enterprise's interfaces to its environment) are missing and eliminate the possibility of mapping transactions beyond company boundaries. One of the major inconveniences of both ontologies - in addition to their increased complexity - is their lack of an adequate graphical representation.

5 Conclusion

The contribution of this paper are five different variants for defining a UML profile for e3-Value. UML is considered as the "lingua franca" in software engineering. However, UML lacks standardized means for describing business models from a value perspective. e3-Value - an approach for value-based requirements engineering - is an ideal complement for designing e-business systems with UML.

When mapping e3-Value to UML we had to consider two somewhat conflicting aspects: On the one hand side, all e3-Value concepts should also be present in a corresponding UML profile. On the other hand side, resulting UML diagrams must be easily understood by business experts with limited UML knowledge. Inasmuch these diagrams must not be overloaded. In our research work, we started off with a mapping towards activity diagrams that include value activities with input/output parameters. At first glance, this solution seemed to be the obvious choice, since the resulting diagram results in the same look and feel as e3-Value diagrams. Unfortunately, not all e3-Value concepts may be mapped in this solution. So we continued with developing alternative variants based on activity diagrams. However, we learned that all the proposed solutions either fail in mapping all e3-Value concepts or result in overloaded diagrams that are hard to read for business people.

Finally, we based the UML profile for e3-Value on use case diagrams. We prefer this solution to the ones based on activity diagrams. It also provides a better integration with the UN/CEFACT modeling methodology (UMM) which serves as a starting point of our research. Being editors of the UMM specification - which is defined as a UML profile for inter-organizational systems - we are looking for an approach that captures the business justification for inter-organizational systems. We believe that the UML profile for e3-Value satisfies these needs. Due to page constraints we were not able to outline the dependencies between UMM and e3-Value. Accordingly, a detailed description of integrating e3-Value into UMM is up to future work.

A UML profile for e3-Value may be used in any UML based software development process and not just in relationship with UMM. Others may prefer a different variant according to their specific needs. Whatever variant is chosen users benefit from our work since they are able to use standard UML tools for modeling e3-Value. We are convinced that this further potentiates the diffusion of e3-Value due to a better integration into the software engineering process.

References

1. Gordijn, J., Akkermans, H.: E3-value: Design and Evaluation of e-Business Models. IEEE Intelligent Systems **16**(4) (2001)
2. Geerts, G., McCarthy, W.E.: The Ontological Foundation of REA Enterprise Information Systems. Technical report, Michigan State University (2000)
3. Osterwalder, A., Pigneur, Y.: An e-Business Model Ontology for Modeling e-Business. In: 15th Bled Electronic Commerce Conf. (2002)
4. UN/CEFACT: UN/CEFACT's Modeling Methodology (UMM), UMM Meta Model - Foundation Module. (September 2006) Technical Specification V1.0, http://www.unece.org/cefact/ umm/UMM_Foundation_Module.pdf.
5. Gordijn, J., Akkermans, H.: Value based requirements engineering: Exploring innovative e-commerce idea. Requirements Engineering Journal **8**(2) (2003) 114–134
6. Gordijn, J., Akkermans, H.: Does e-Business Modeling Really Help? In: Proc. of the 36th Hawaii Intl. Conf. On System Sciences, IEEE (2003)
7. Buhr, R.J.A.: Use Case Maps as Architectural Entities for Complex Systems. IEEE Trans. Softw. Eng. **24**(12) (1998)
8. Object Management Group (OMG): Unified Modeling Language: Superstructure. (February 2007) Version 2.1.1, http://www.omg.org/docs/formal/07-02-05.pdf.
9. Russell, N., van der Aalst, W.M., ter Hofstede, A.H., Wohed, P.: On the Suitability of UML 2.0 Activity Diagrams for Business Process Modelling. In: 3rd Asia-Pacific Conf. on Conceptual Modelling, Australian Computer Society, Inc. (2006)
10. Gordijn, J., Akkermans, H., Van Vliet, H.: Business Modelling is not Process Modelling. In: ER '00: Proc. of the Workshops on Conceptual Modeling for E-Business and the Web, Springer LNCS (2000)
11. Penker, M., Eriksson, H.E.: Business Modeling With UML: Business Patterns at Work. Wiley (2000)
12. McCarthy, W.E.: The REA Accounting Model: A Generalized Framework for Accounting Systems in a Shared Data Environment. The Accounting Review **57**(3) (1982) 554–578
13. Hruby, P.: Model-Driven Design Using Business Patterns. Springer (2006)
14. Osterwalder, A.: The Business Model Ontology. A Proposition in a Design Science Approach. Dissertation, University of Lausanne (2004)
15. Andersson, B., Bergholtz, M., Edirisuriya, A., Ilayperuma, T., Johannesson, P., Gordijn, J., Grégoire, B., Schmitt, M., Dubois, E., Abels, S., Hahn, A., Wangler, B., Weigand, H.: Towards a Reference Ontology for Business Models. In: Proc. of the 25th Int. Conf. on Conceptual Modeling, Springer LNCS (2006)
16. Weigand, H., Johannesson, P., Andersson, B., Bergholtz, M., Edirisuriya, A., Ilayperuma, T.: On the Notion of Value Object. In: Proc. of the 18th Intl. Conf. on Advanced Information Systems Engineering, Spring LNCS (2006)
17. Uschold, M., King, M., Moralee, S., Zorgios, Y.: The Enterprise Ontology. The Knowledge Engineering Review **13**(1) (1998) 31–89
18. Fox, M., Gruninger, M.: Enterprise Modelling. AI Magazine **19**(3) (1998) 109–121