e3 VALUE

Jaap Gordijn
Roel Wieringa

DESIGNING YOUR ECOSYSTEM IN A DIGITAL WORLD

THE ULTIMATE BUSINESS MODELING LANGUAGE

USER GUIDE

E^3value User Guide

Designing Your Ecosystem in a Digital World

Jaap Gordijn and Roel Wieringa

***E*^{3}*value* User Guide**
Designing Your Ecosystem in a Digital World

Edition:
2021 02 12: First release
2023 05 10: Second Release

Cover art: `http://borinka-and-shamrock.com`

Published by:
The Value Engineers B.V., Soest, NL
contact@thevalueengineers.nl
Soest, The Netherlands
KvK: 69731101

ISBN 978-90-828524-4-8

See `https://www.thevalueengineers.nl/` for software tools for *e*^{3}*value* and for the pdf version of this book.

Contents

Preface

This is book treats the syntax and semantics of e^3*value* and explains the various analyses that can be done with the e^3*value* software tools. E^3*value* is a language to build and analyse business models of value networks.

Software tools to edit and analyze e^3*value* are available from our web site www.thevalueengineers.nl. Currently there are four tools.

- The e^3*value* tool allows editing and analysis of e^3*value* models according to the techniques described in this book
- The e^3*value* time series editor (downloadable together with the e^3*value* tool) allows the creation and analysis of time series according to the techniques described in this book.
- The e^3*tool* allows editing and analysis of e^3*fraud* models.
- The e^3*web* tool runs in any browser window and currently only contains editing functionality for e^3*value* models.

Eventually, the functionality of all tools will be implemented in the e^3*web* tool, and the older tools will be phased out. Currently, all tools offer some interesting functionality and this book describes all of them.

This book is available as pdf file, as set of help pages in the e^3*web* editor and on our web site, and as a paper book in online bookshops.

To be neutral with respect to regions where this book is used, we use $f$as generic currency symbol. Pronounce: “Florin”.

Chapter 1

Introduction

1.1 Value networks

Network organizations have existed as long as businesses have existed. Every business, no matter what technology it uses, interacts with a network of suppliers, customers, partners, sponsors, regulators, malicious actors, competitors, and other actors. For example, Ikea is part of a value network of designers, manufacturers, suppliers of raw materials, logistics companies and others who cooperate to deliver furniture to customers. Without its value network, Ikea has nothing to offer. Hence, we view its offer of furniture as an offering made by its entire value network.

The concept of a *value chain* is not sufficient to describe this network, because the actors with which a company interacts do not form a linear chain. Other terms that have been used are *network organization, joint venture, modular corporation, value-adding partnership, virtual corporation, extended enterprise* and *value web* [1, 4]. We use the term *value network.*

1.2 Innovation

If every business operates in a value network, then how can a business innovate? **Innovation** is changing the way you create, deliver or capture value. Businesses as well as non-profit organizations can innovate by using new technology, processes or organization structures for creating, delivering or capturing value.

But if every business is embedded in a network, then business innovation is embedded in network innovation. When a business innovates, it may use new technology, processes or organization structures in the interactions with other actors in their value network. It may select new suppliers, other partners, acquire new customers, and compete with new competitors. It may be subject to different regulations and may have to deal with new standards. This means that to implement an innovation, it must innovate its value network.

This is not a new problem. It has existed as long as businesses exist. However, in today's world of complex value networks, high-speed communication and global logistics, innovation requires a major effort to understand your value network and to convince independent, profit-and-loss-responsible actors to go along with you. This is where *e³value* is useful.

1.3 Value models

E³value is a language and set of analysis techniques to represent and analyse value networks. A **value model** is a representation of a value network in *e³value* .

A value model does not represent processes but economic exchanges in which two or more parties exchange something of value for them. It does not represent when and how these exchanges take place. It represents

- who exchanges what value objects with whom;
- which customer needs are answered by this, and

- what revenue and expenses are generated by this for each actor.

A value model represents a value network during a period of time, called the **contract period.** The exchanges among actors represented in the value model are agreements about what objects of economic value the actors will exchange during the contract period.

e³value represents just enough to be able to make revenue estimations and net present value estimates for actors in the network. To do this you annotate a model with estimates of market size, number of customer needs and the monetary value of object exchanges among actors. Our supporting tools can then compute assessment of commercial viability and well-being for a contract period, and can do net present value flow estimations for a sequence of contract periods.

1.4 When to use *e³value*

We use *e³value* to understand existing value networks and to design new ones.

Existing value networks may contain inefficiencies and vulnerabilities. For example you may know who your suppliers are, but not who the suppliers of your suppliers are, which creates a vulnerability. You may engage in economic exchanges with other parties in a way that is not optimal for your current value proposition. And you may coordinate your work with others, or *fail* to coordinate your work with others, in a way that destroys potential value. Modeling an existing value network allows you to identify and fix these inefficiencies.

To realize a new business idea, you need to design a *new* value network. You can represent the design of a new value network in *e³value* and simulate various market scenarios with our supporting software tools. This allows you to quickly identify strong and weak points of an innovation without losing money or hurting others.

The *TVE business model template* contains more information

about what a network-aware business model looks like[1] The TEAM method for ecosystem modeling provides guidelines for modeling business ecosystems in general and value networks in particular.[2] We will explain these approaches in a later book.

[1] https://www.thevalueengineers.nl/the-structure-of-business-models-an-update/

[2] https://www.thevalueengineers.nl/what-is-your-ecosystem-the-ecosystem-architecture-modeling-framework/

Chapter 2

Value Networks

A **value network** is a collection of independent actors offering something of value to a market by engaging in economic exchanges.

The actors in a value network may be

- profit-and-lost-responsible companies, or they may be
- nonprofit or government organizations, or
- individual consumers.

Members of the network may be partners or competitors of each other. The characteristic features of a value network are that its actors are economically independent, engage in economic exchanges, and jointly offer something of value to their environment.

A value network consists of *actors,* including the customer who consumes the offering of the network. Actors may be grouped in *market segments.* Each actor performs one or more *value-adding activities.* The actors are related by *commercial transactions.*

In this chapter, we explain these concepts and illustrate how they are modeled in e^3value .

2.1 Actors

An **actor** is an entity that is responsible for its survival and well-being. This means that it is perceived by itself and its environment as economically independent. More specifically,

- businesses are profit-and-loss responsible,
- non-profit institutions and government organizations are responsible for acquiring a budget and complying to their mission, and
- consumers are responsible for their own survival and well-being.

Most actors are legal or biological persons. However, a profit-and-loss-responsible business unit is an actor too — it is responsible for its survival and well-being— but it is not a legally independent entity.

In reality, actors have *relationships* with each other that help them survive and thrive. Companies form alliances and they may have ownership relations. Non-profits have cooperation relationships. Individual consumers are part of a network of all kinds of social relations. And all actors have competitive relationships with other actors. Of this vastly complex network of relationships, e^3value represents only type: that of economic transactions. This is a simplification and abstraction that makes it possible to understand one aspect, that of economic transactions, of the complex of networks that we are all part of.

In an economically viable network, all actors can survive. In a healthy network, they all thrive. E^3value is designed to analyze economic viability and health of a network in the long run.

Actors are represented in e^3value by rectangles, as illustrated in figure 2.1. The name of an actor expresses its role in a value network (Traveler, Railway company) or it is a proper name (Deutsche Bahn).

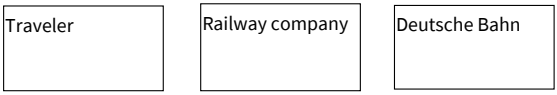

Figure 2.1: Representation of actors in *e³value* . Two actors are named by role, one by its proper name.

2.2 Market segments

A set of actors that assign value to objects in the same way may be grouped into a **market segment**. For example, the set of subscribers of Netflix may be grouped into a market segment, because from the point of view of Netflix, they all attach the same value to streaming video.

Of course, in reality, different customers may attach a different value to streaming video. But Netflix offers all its customers a choice among the same subscription schemes and subscribers pay the same amount based on their subscription scheme. The concept of market segment is a modeling abstraction that helps us understand a value network.

Market segments are represented in *e³value* as three stacked actors (figure 2.2). A market segment has a name unique in the *e³value* diagram. We recommend the name to be plural form, such as Customers or Artists.

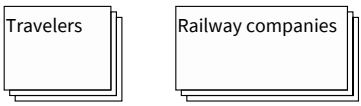

Figure 2.2: The market segments of Travelers and Railway companies.

Two market segments may have a non-empty intersection. For example, a movie actor can have a subscription to Netflix streaming

Figure 2.3: One person may be both a Movie actor and a Netflix customer. This overlap in market segments is not represented in *e³value* .

video (figure 2.3). The actor is then part of two market segments, that of Movie ctors and of Netflix customers. An overlap of two market segments is *not* represented in *e³value* .

Another kind of double appearance is when a business actor is part of a market segment and also appears as individual actor in the diagram. For example, Facebook appears as an individual business actor in the value network of Facebook. Facebook places advertisements for itself on all social networks, including itself. So the Facebook value network contains a market segment Social Networks, that contains Facebook itself (figure 2.4). Again, we do not represent these relations in *e³value* .

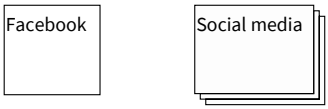

Figure 2.4: A business can appear as independent actor and be also part of a market segment. This overlap is not represented in *e³value* .

Size of market segment. A market segment has a size, which is the number of actors in the segment.

2.3 Value activities

A **value activity** is a task performed by an actor that potentially results in a benefit for the actor.

- If the actor is an enterprise, a value activity potentially results in a *positive net cash flow* and the value activity is a profit center.
- If the actor is a non-profit or government institution, a value activity potentially contributes to the organization's *mission.*
- If the actor is an individual, the value activity potentially contributes to an increase in *economic utility* for the individual.

A value activity is represented in *e*^{3}*value* by a rounded box (figure 2.5). We recommend to use the present continuous tense (e.g. traveling) for the name of value activities.

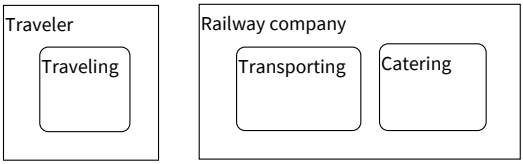

Figure 2.5: The railway company performs transporting and catering services.

If no value activity is shown in an actor, this means that it performs exactly one value activity, which is left implicit. We should be able to infer the value activity of the actor from its name. If we would omit the Traveling activity fro Traveler in figure 2.5, we can still guess what a traveler does: traveling.

A market segment can perform a value activity too, which means that each actor in the market segment performs the same value activity (figue 2.6).

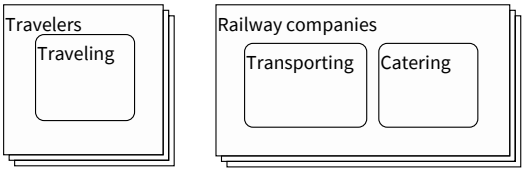

Figure 2.6: Actors in a market segment perform the same value activity.

Sourcing. An important task in value network design is the assignment of value activities to actors, also called *sourcing*. A value activity is a collection of operational activities that can be assigned as a whole to actors. A value activity should be assigned to an actor who expects to benefit from performing the value activity, i.e. to make a profit, comply to its mission, or increase economic value for itself. From the point of view of one actor, the assignment of a value activity to an actor or market segment is basically a make-or-buy decision.

For example, in figure 2.5, the railway company offers two services, Transporting and Catering. In figure 2.7, the Catering activity has been outsourced. In this situation, the catering company expects to profit from this activity. It can provide its catering services also to other actors.

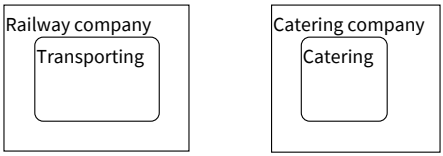

Figure 2.7: The railway company has outsourced catering services.

2.4 Partnerships

Actors may cooperate in various relations to offer something of value to customers. Examples are revenue sharing, cross-selling, franchising, and many more [7]. *E³value* only represents transactions among actors, no contractual relations. However, if two or more actors offer their products as a bundle to customers, we say that they have a **partnering** relationship. This is modeled by a *composite actor,* which is represented by a rectangle around the actors whose products are bundled (figure 2.8).

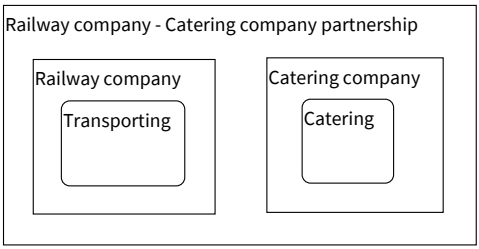

Figure 2.8: The railway company and the caterer form a partnership to offer their services as one bundle.

The partnership is *not* the owner of the participating actors. Partnerships are only used to represent bundling of products of independent actors. In figure 2.8, the Railway company and the Caterer form a partnership in which they offer their services in one bundle. The Traveler cannot observe that the service is offered by two independent actors, and hence for the Traveler the situation is indistinguishable from that of figure 2.5.

Chapter 3

Economic Transactions

In an economic transaction, two or more actors exchange value objects to satisfy a need that one of them has or to produce other value objects that contribute to need satisfaction of another actor. We spell this out using the simple, incomplete example of figure 3.1.

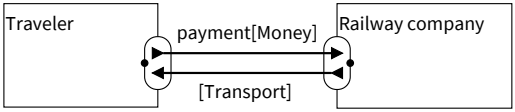

Figure 3.1: Traveling from Amsterdam to Paris. This model is used to explain the concepts introduced in this chapter.

3.1 Value objects

A **value object** is a money- or non-money object that

- is of economic value for at least one other actor in the network and that

- satisfies a need directly or indirectly (through another value object).

Value objects should be countable, which means that it is meaningful to ask how many of them there are. Value objects have economic value, defined differently for money- and non-money objects.

A *money object* has a definite value that is known and agreed on by actors. If actor A pays $f10$ to actor B, then B receives $f10$ from A.[1] Money objects are also called *value-in-transfer* [6].

A *non-money object* may be a physical good (of the kind that you can touch or can drop on the floor), the outcome of a service, or an experience. The value of a non-money object for an actor depends on the increase in utility when the actor consumes or uses the value object. Different people may assign different value to non-money objects, and one person may even assign a different value at different times. For example, the value of a cold beer to me on a hot summer day is different from its value on a cold rainy day. The value of an non-money object is often called *value-in-use* [6].

A value object is represented by showing the name of the object next to a value transfer enclosed in square brackets (as [value object]). In figure 3.1, two value objects are represented: Money and Train trip. The name of a value object should express the economic utility of the object. It should explain why an object is of value to an actor.

All value objects have a valuation, which for money objects expresses the value in some currency. In this book we represent that currency by the symbol f, pronounced "Florin".

3.2 Value ports

A **value port** is a willingness to *provide* or *request* value objects. Value ports belong to actors, market segments or value activities.

[1] To avoid examples in a particular currency, we use $f$as a neutral money symbol. Pronounce "Florin".

- A value port of an *actor* is a willingness to provide or request value objects to or from its environment.
- A value port of a *market segment* is a willingness of each actor in the segment to provide or request value objects to or from its environment.
- A value port of a *value activity* of an actor is a willingness of the activity to provide or request value objects to or from its environment inside or outside the actor.

Each value port has a **role** *in* or *out*. This indicates whether the actor requests or is willing to provide a value object, respectively. A value port is depicted by a small triangle that indicates the role of the port (figure 3.1).

3.3 Value transfers

A **value transfer** is a willingness of a provider and a requester to transfer a value object from the first to the second. Providers and requesters can be actors, market segments, or value activities.

A value transfer is represented by a line connecting the out-port of a provider with the in-port of a requester, labeled with the name of the transferred value object. Figure 3.1 shows a transfer of a Train trip fro the Railway company to the Traveler and a payment of Money in the reciprocal direction.

A value transfer may additionally be labeled by the name of the transfer. The transfer name should express the role of the transfer for the actors connected by the transfer. For example, In figure 3.1 the transfer of Money has role payment.

A value transfer occurrence is an event in which an instance of the value object is transferred from provider to requester. All value transfer occurrences are discrete events that can be counted.

This is an economic fiction because physically, there is always a process. For example, provision of a the train trip from Amsterdam

to Paris leaving Amsterdam at 8:42 on the first of April, 2020, is a process that takes a few hours. Provision of electricity to a home during the month of April, 2020 is a continuous occurrence that lasts a month.

The economic fiction is easier to understand if we realize that a value transfer occurrence is not a physical process but the *transfer of a right.*

- For money and physical goods, the transferred right is ownership.
- For services, the transferred right is the right on the outcome of the service.
- For experiences, the transferred right is the right to participate in the experience.

3.4 Value interfaces and value transactions

A **value interface** groups incoming and outgoing value ports into an atomic value exchange. A value interface must contain ports of opposite roles, i.e. at least one in-port and one out-port. An interface containing only outgoing or only ingoing ports is impossible. Actors, market segments and value activities may all have value interfaces. We discuss actor value interfaces in this section and interfaces of market segments and activities in later sections.

A value interface is visualized by a rounded box around its ports. Figure 3.1 shows two value interfaces.

Reciprocity. Value interfaces express *economic reciprocity*, which is the property that a rational actor is only willing to offer objects to another actor if it receives adequate compensation. Usually the compensation for a non-money value object is a reciprocal money object, but this is not a requirement. The reciprocal object may be

any value object that the requestor perceives to increase its economic utility.

Atomicity. Transfer of value objects across a value interface is atomic. In an occurrence of a value transfer of a value interface, either all transfers of the value interface occur in the contract period, or none of them is. This ensures that if an actor offers something of value to someone else, it always gets in return what it wants in the same contract period. *How* this is ensured is a matter of a robust business process design, trust and associated control mechanisms, legal agreements, or technology. This is not expressed in the value model.

Also, value models do not express an ordering of value transfer occurrences. For example, in figure 3.1, the traveller may first pay and make the trip a month later. Or the traveler may pay when entering the railway station, or when entering the train, or pay at the end of the month in which the trip was made. Timing and ordering of actions is not modelled by an e^3value model.

What *is* modeled is that if a value transfer of an interface occurs in the contract period of the model, then all other transfers of the interface also occur in the contract period. This is precisely enough to count the number of occurrences of value transfers, e.g. of payments, and do net value flow computations for a contract period.

Value transaction. A **value transaction** is the set of value transfers triggered when a value interface is triggered. We will see that value transactions can involve more than two actors and more than two interfaces.

3.5 Value offering and bundling

The collection of ports with the same role in a value interface is called a **value offering**. Each value interface has exactly one out-going offering and one in-going offering.

For an offering paid for in money, we need a *pricing model* that expresses the price of the offering in quantitative terms. The price may be fixed per offered value object, or per time period (subscription model). It may be based on usage or on performance of the supplier. The design of a pricing model is a crucial decision when desiging a value interface [7].

Bundling. Value offerings can be used to express *bundling.* Bundling can be motivated by the supplier and/or by the customer. In *supplier-side bundling*, an actor wants to offer value objects in combination rather than separately, because it thinks that different products sold in combination yield more profit than that if they were sold separately [2]. Figure 3.2 shows an example.

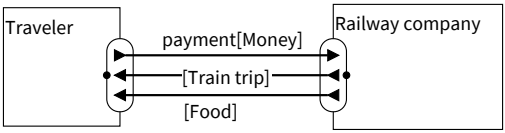

Figure 3.2: The offering of the railway company bundles food with a Train trip. The traveller has no other choice than to buy this bundle.

Supplier-side bundling can be less coercive for the customer than this. For example, McDonalds' Happy Meal is a bundle of various software products that can also be obtained separately, apart from the toy. However, the pricing of the Happy Meal is such that the purchase of the whole bundle is encouraged, rather than a few of its components.

Customer-side bundling occurs if a customer requests two or more value objects in combination in one value interface (figure 3.3).

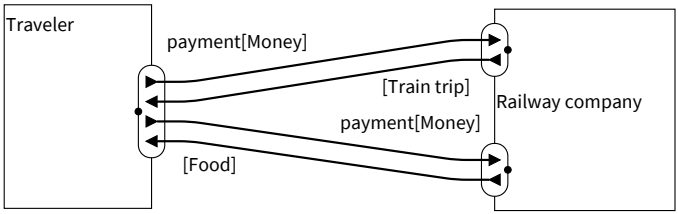

Figure 3.3: The Traveler bundles a Train trip with Food. These two value objects only have utility for the Traveler in combination.

3.6 Value interfaces of market segments

A value interface of a market segment expresses that all actors in the segment offer the same value interface to their environment. Figure 3.4 gives an example.

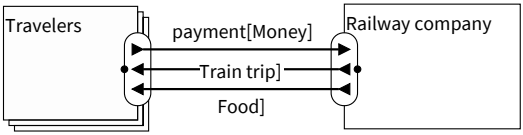

Figure 3.4: All actors in the Traveler market segment offer the same interface to their environment.

3.7 Value interfaces of value activities

Value activities have value interfaces too, as illustrated in figure 3.5. In this model, the traveler buys the trip and food in one bundle but the railway company sells train trips and food separately.

As an alternative, the railway company can choose to sell food only in combination with a train trip. This is represented in the

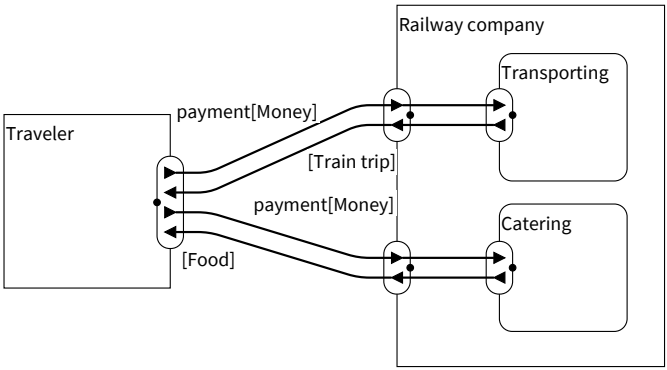

Figure 3.5: Two value activities that contribute to two offerings of the Railway company.

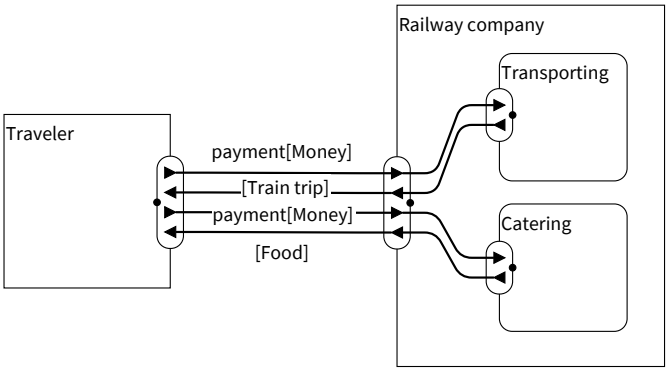

Figure 3.6: Two value activities that contribute to the same offering of the Railway company.

model of figure 3.6. This is an example of bundling at the supplier as well as customer side.

The railway company may alternatively decide that it is in the luxury transporting business and will offer catered train trips. The traveler pays one price for the trip and food. The luxury rail transporting activity buys this food from the catering value activity (figure 3.7). The catering activity must be paid for this because otherwise it would not be a profit center.

If the railway company does not think catering can be a profit center, it can outsource it, as shown in figure 3.8. Perhaps catering is profitable for a specialized catering company.

3.8 Value interfaces of partnerships

The railway can alternatively decide to cooperate in a different way with a catering company. In a partnership they offer a joint interface to the customer and the railway company itself has no transactions with the catering company. Rather, the partnership has some operational activities to operate this joint interface. These activities are not a profit center and so are not shown in the value model (figure 3.9).

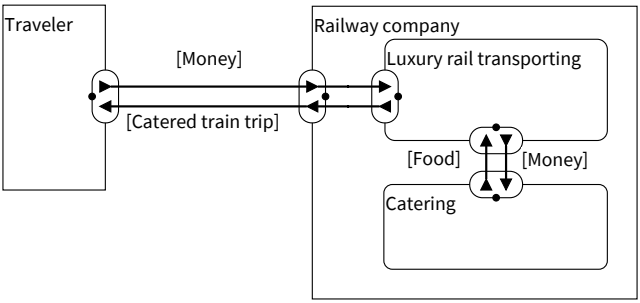

Figure 3.7: An internal profit center.

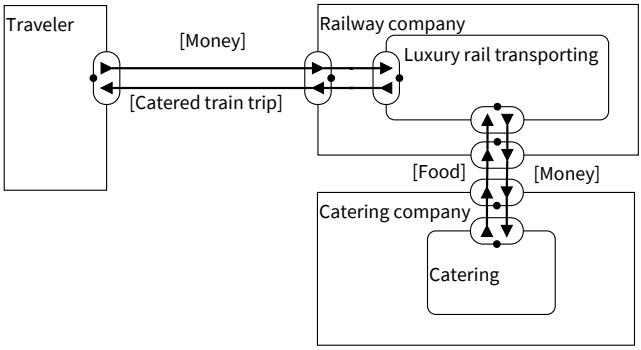

Figure 3.8: An outsourced activity.

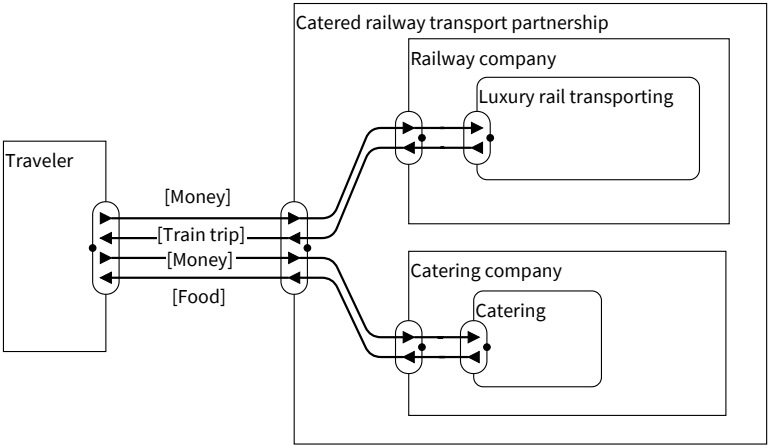

Figure 3.9: Joint interface through a partnership.

Chapter 4

Dependency Elements

Actors and value activities have one or more value interfaces through which they perform economic transactions with the outside world. There are internal dependencies among the interfaces of each actor that must be included in a business model of a value network.

For example, to provide a luxury rail transport service, a railway company has to acquire food somewhere to serve during the trip. The value activity Luxury rail transporting thus has two value interfaces, one for providing the luxury rail transport service and the other to acquire food. The first interface depends on the second. These dependencies are represented in an *e*^{3}*value* model by means of **dependency elements**.

Dependency elements do *not* represent the time ordering of transactions nor the operational activities of actors. Dependency elements represent the value-adding logic of an actor by relating the value interfaces of an actor to each other, to customer needs, and to boundary elements that indicate the scope of the model.

4.1 Customer needs

A **customer need** is a lack of something valueable that the actor wants to acquire.

- For profit-and-loss-responsible actors, it is the need to generate sufficient revenue to survive and thrive.
- For non-profits and government institutions, it is the need to be compliant with their mission.
- For consumers, it is a state of felt deprivation of some basic satisfaction, also called *consumer need* [5].

A customer need is visualized by a bulls-eye connected to the interfaces needed to satisfy this need (figure 4.1). In the market segment, each of the travelers in the segment has the need to travel.

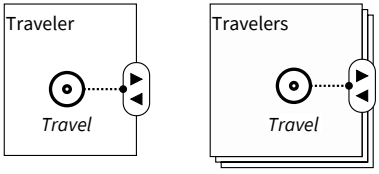

Figure 4.1: An actor and a market segment with a need.

Needs are goals, and the name of a need should express this goal. We recommend choosing a name that prefixed with "Need to" yields a meaningful noun. For example, a Traveler has the *need to travel*, which makes "Travel" a good name for the need.

To understand a value network, we must understand the customer need it aims to satisfy. Jointly, the actors in a value network offer a value object to a customer. When the customer acquires value objects to satisfy its need, a collection of economic transactions in

the network is triggered. It is the performance of these transactions by which the network can survive and thrive.

4.2 Boundary elements

A **boundary element** represents the limit of the scope of a value model. It is represented by a cross connected to the interfaces whose business logic we are not going to analyze further (figure 4.2).

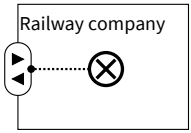

Figure 4.2: A boundary element.

In figure 4.3, the interface of the Catering activity is connected to a boundary element, which means that the model does not show how the value activity is able to offer its service to the Luxury rail transporting activity. The Catering activity needs to acquire the food from somewhere and needs to buy equipment to prepare it. So in reality it has additional interfaces to deal with this. The boundary element indicates that our model ignores these interfaces.

In addition, the railway company needs to buy rail cars, buy fuel to drive the cars, buy or rent space on railways, and acquire the right to transport passengers. All of these activities have a value interface to the environment of the company. Our model ignores them.

If we were interested in the long-term economic viability of the railway company, we would have to include all of these interfaces. In addition, in that extended model we would include actors that sell rail cars, provide fuel, provide space on rail tracks, etc. If we are not interested in the economic viability of those actors, we would place boundary elements in those actors.

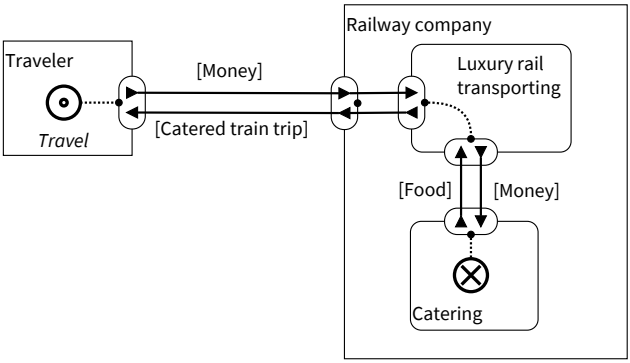

Figure 4.3: A dependency graph.

Whatever interfaces we include, there will always be many interfaces that we have omitted. Boundary elements are used to limit the scope of the model. The decision where to put the model boundary depends on our *modeling goal:* If we want to analyze the economic sustainability of an actor, we model all its value interfaces. If we are not interested in the economic sustainability of an actor, we ignore some of its interfaces and put a boundary element in the actor.

4.3 Dependency paths

A **dependency path** is an acyclic graph starting with one or more customer needs and ending in one or more boundary elements. The meaning of a dependency path is that if one of its roots is triggered in a contract period, all value interfaces through which the path passes are triggered in the contract period. This is all we need to know to compute the value flows generated by a customer need.

Mathematically, a dependency graph is a rooted acyclic graph. It may have more than one root, and all its roots are customer needs.

It may have more than one boundary element as terminal. There must be no cycles in a dependency graph, because otherwise there could be infinitely many transactions triggered by one occurrence of a customer need.

Dependency paths abstract from all operational activities that an actor performs to request and provide value objects. In particular, a dependency graph abstracts from all *time* ordering of transactions and merely connects all value transactions that occur to satisfy a customer need.

4.4 And-dependencies

An **and-dependency** is a logical conjunction of dependencies. It is represented by a line with a *connection point* on one side that represents the *conjunction*, and connection points on the other side that represent the parts of the conjunction, called the *conjuncts* (figure 4.4). This is a purely logical construction and represents no time ordering. So in figure 4.4, $a = (b \text{ and } c \text{ and } d)$. By this we mean that

> *a occurs in the contract period of the model if and only if b, c and d occur in the contract period of the model.*

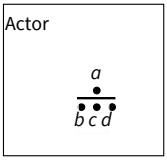

Figure 4.4: An and-dependency.

A dependency path may contain and-dependencies.

For example, in figure 4.5, the traveler need occurs in the contract period if and only if both traveler interfaces are triggered in the contract period.

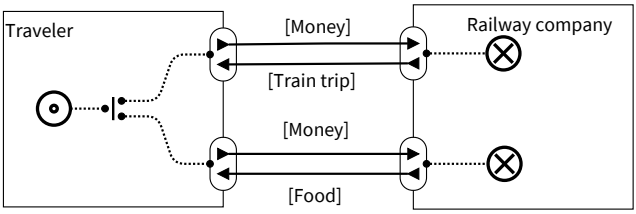

Figure 4.5: An and-dependency: both a train trip and food are needed to satisfy the consumer need.

In figure 4.6, the Luxury rail transporting activity has outsourced the Catering and Ticketing activities in order to be able to concentrate on its core competency, providing luxury rail transport.

4.5 Or-dependencies

An **or-dependency** is logical disjunction of dependencies. It is a represented by a small tree with a connection point at its root that represents the disjunction and connection points at its leaves that represent the parts of the disjunction, called the *disjuncts.* For example, in figure 4.7, $a = (b \text{ or } c \text{ or } d)$. By this we mean that

> *a occurs in the contract period of the model if and only if exactly one of b, c or d occurs in the contract period of the model.*

A dependency path may pas through an or-dependency.

For example, in figure 4.8, the Luxury transporting value activity chooses between two suppliers of food. For each occurence of the customer need, one of the two suppliers supplies the food. How and

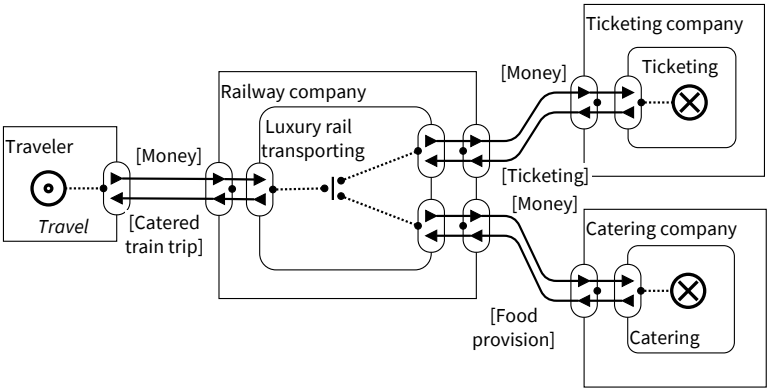

Figure 4.6: An and-dependency that represents outsourcing of two activities.

when these choices are made is not represented. The choice may be made once per year, once per month, based on external conditions, per group of travelers, or in other ways. A value model does not provide process information.

We can construct complex and-or logic by combining and- and or-dependencies. In figure 4.9, each time the traveler need occurs, the traveler makes the choice whether or not to combine the purchase of a train trip with the purchase of food. The upper choice in the diagram leads to a purchase of a train trip only. The lower choice leads to a purchase of a train trip *and* of food.

4.6 Dependency graphs

Due to or-dependencies, there can be more than one dependency path starting from a customer need, one for each branch of the or-dependency. A **dependency graph** is the set of all dependency paths starting from the same needs. Figure 4.9 is an example.

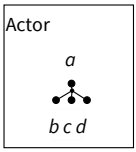

Figure 4.7: An or-dependency.

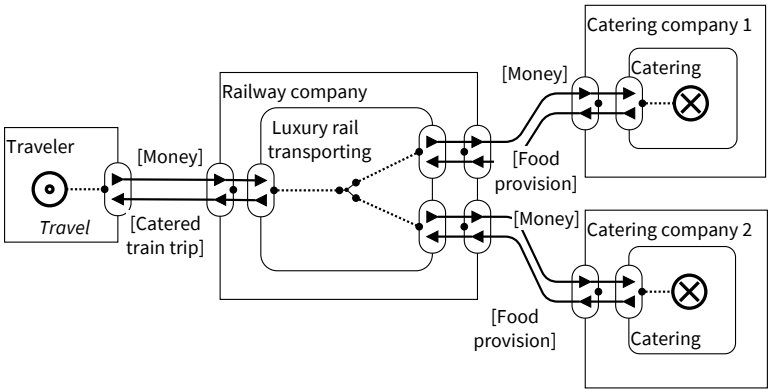

Figure 4.8: An or-dependency that represents a choice in suppliers.

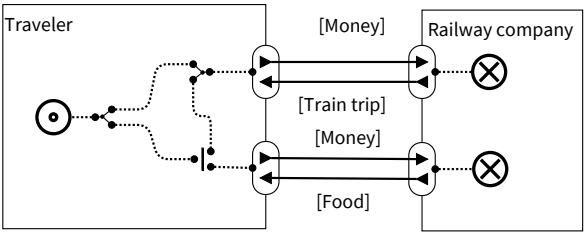

Figure 4.9: OR dependencies.

Chapter 5

Quantification

This chapter describes the quantification capabilities of the *e³value* tool. This has not yet been implemented in the *e³web* tool.

To quantify an *e³value* model, you specify how many times value transfers occur in a contract period, and what the value of the transfered value objects for the participating actors is. We call a quantification of the model elements a **market scenario.** Remember that a contract period is the period of time represented by an *e³value* model. A market scenario contains the quantifications for one contract period.

If the *e³value* model represents an *existing value network,* the numbers in a market scenario can be based on historical data. If the *e³value* model represents a possible *future network,* the numbers in a market scenario are estimates of future behavior. In both cases, modeling existing or new business ideas, quantification is a great way to analyze and debug the the model. In addition, to test sensitivity of a value network to differences in market conditions, you typically create several market scenarios to see how the value network behaves under different conditions.

To count value transfers and assign value to transferred value objects, we need to set properties of some of the elements of a model.

We use the model of figure 5.1 as running example in this chapter.

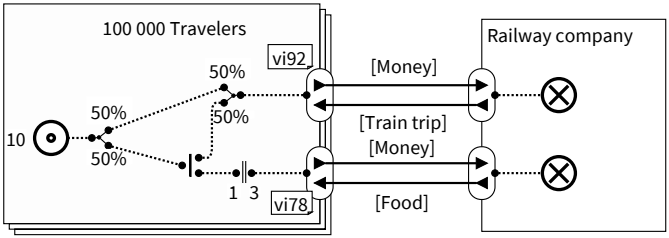

Figure 5.1: Running example. 100 000 Travelers each have 10 travel needs, in 50% of which they buy food three times. The labels "vi92" and "vi78" in the comments are arbitrary labels assigned by the editor to the value interfaces of Travelers.

5.1 Properties

E^3*value* model elements have pre-defined properties that can have a value. For example, most model element have a predefined **name**. Appendix B lists all pre-defined properties of all model elements.

You can set and change some of the predefined properties. Other predefined properties are derived by the editor and cannot be changed by you. This is defined in appendix B and indicated in the editor

In addition to the predefined properties, you can add as many other properties to an element as you wish.

For simplicity, all our examples here use constant property values, except in section 5.11, where we give examples of computed values.

5.2 Customer need occurrences

During a contract period, there will be a number of **occurrences** of a need. If a customer need is assigned to a market segment, this means that on the average, each of the actors in the segment has that many occurrences of the need in the market scenario.

In figure 5.1 , we have defined the number of Traveler needs to be 10. This means that on the average, for each Traveler in the Travelers market segment the need to travel occurs 10 times in this scenario.

5.3 Sizing market segments

Market segments have an **size** property that indicates the number of actors in the segment. The total number of need occurrences in the contract period for that segment is

$$occurrences_{need} \times size_{marketsegment}.$$

The market segment of Travelers has size 100 000. This means that in this scenario, the need to travel occurs 10 * 100 000 = 1 000 000 times.

5.4 Quantifying or-dependencies

Each disjunct of an or-dependency has a ratio, which is used to compute the number of occurrences of a disjunct in a market scenario.

Going along the dependency path from the customer need to the boundary elements in figure 5.1, we encounter first an *or-split* and then an *or-join.* At the or-split, the 10 consumer needs are split equally over the two choices. So in this scenario, 5 times the Traveler buys a Train trip without food, and 5 times the Traveler buys a trip with food.

The ratios of the disjuncts of the or-join must match the number of occurrences that enter the join.

5.5 Cardinality dependencies

Sometimes, triggering a need triggers a value interface more than one time. This can be modeled by a **cardinality dependency**, which is represented by a pair of parallel lines each with a connection point that has a cardinality.

For example, in figure 5.1, each need for food results in three value transfers of Food (and consequently of Money). The cardinality dependency is triggered 5 * 100 000 times, so the value interface vi78 is triggered 15 * 500 000 = 7 500 000 times.

The default cardinalities of a cardinality dependency are 1, in which case it can be omitted from the diagram.

A cardinality dependency represents a jump in cardinality. Cardinality jumps may be up or down (figure 5.2). A jump to a higher value is called an **explosion**, a jump to a lower value is called an **implosion**.

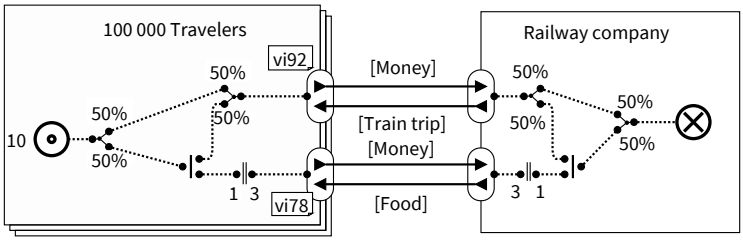

Figure 5.2: Implosion element, and- and or-joins.

Figure 5.2 also illustrates that we can match an and- and or-splits with corresponding joins. This allows us to continue the dependency graph as a single path, in this case to a single boundary element.

5.6 Same-object bundling

It often makes business sense to sell a bundle of non-money object occurrences of the same type at the same time. For example, the Railway company may offer a bundle of 3 meals in one sale. This makes no sense for money objects (the buyer will only pay once for an offering) but for non-money objects this is a common phenomenon.

We can specify this by placing a cardinality at a value port or at a value transfer of non-money value objects. We place the cardinality at the port if the bindling is defined by the seller or buyer. We place it at the transfer if it is the result of an agreement between seller and buyer.

5.6.1 Port cardinalities

A **value port cardinality** specifies how many instances of the value object are transferred through the port when the value interface is triggered. An *occurrence* of a value port is an event in which it accepts or provides a value object. The *cardinality* of a port indicates how many instances of the value object are transferred in each occurrence.

For example, in figure 5.3, the Railway company offers meals in a bundle of three.

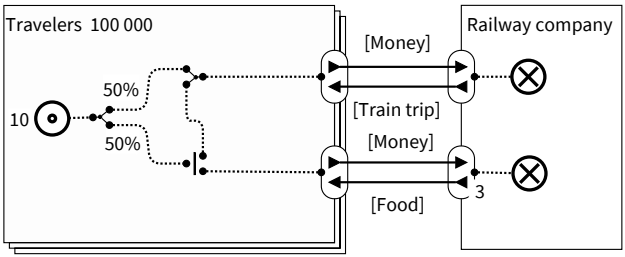

Figure 5.3: A supplier-side port cardinality.

If bundling were defined by the Travelers, and not by the Railway company, then we should have put it at the in-port of the customer. For example, in figure 5.4 the Travelers all buy a bundle of three meals from the Railway company even though the company offers them separately.

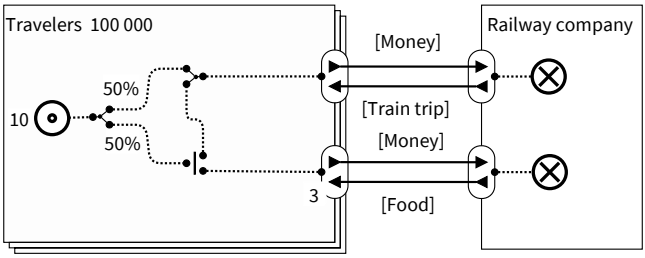

Figure 5.4: A customer-side port cardinality.

By default, a port cardinality is undefined, in which case the algorithm to count value transfers will ignore it. When the editor analyzes a model, it checks that port cardinalities are not set at both ends of a transfer.

If a port-cardinality is set, it must be connected to a transfer and it must be set at one side of the transfer only.

5.6.2 Value transfer cardinalities

If provider and requester have agreed on a bundle, we put the cardinality in the transfer. The *cardinality* of a transfer indicates how many instances of a value object are transferred in one occurrence of the transfer. See figure 5.5.

By default a transfer cardinality is undefined. If it is defined, it overrides the cardinality of the ports to which it is connected.

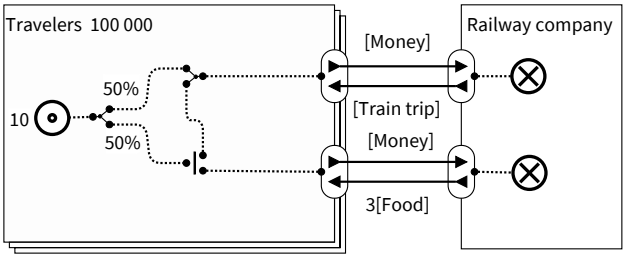

Figure 5.5: A transfer cardinality.

5.6.3 Where to specify same-object bundling?

We now have four ways to specify same-object bundling: As a cardinality jump, as a port cardinality (two sides), and as a value transfer cardinality. Specifying it as a cardinality jump expresses a property of the internal value activities of an actor. Specifying it as a port cardinality expresses customer-side or supplier-side bundling. Specifying it at a value transfer expresses an agreement between supplier and customer.

5.7 Transaction choice ratios

Figure 5.6 shows that the Business Traveler has a choice at each of its input and output ports. For example, it may buy 5 Train trips and 5 meals from Railway company 1. However, the logic of the choices allows three other transactions, not all of which are intended.

For example, the business Traveler may buy Train trips from Railway company 1 and food from Railway company 2. This is not intended but the diagram allows it. To rule it out, we need a transaction table.

In a *transaction table*, we indicate which transactions are intended and in which ratio they occur. The table is part of the e^3value model. For example, table 5.1 shows which transactions are

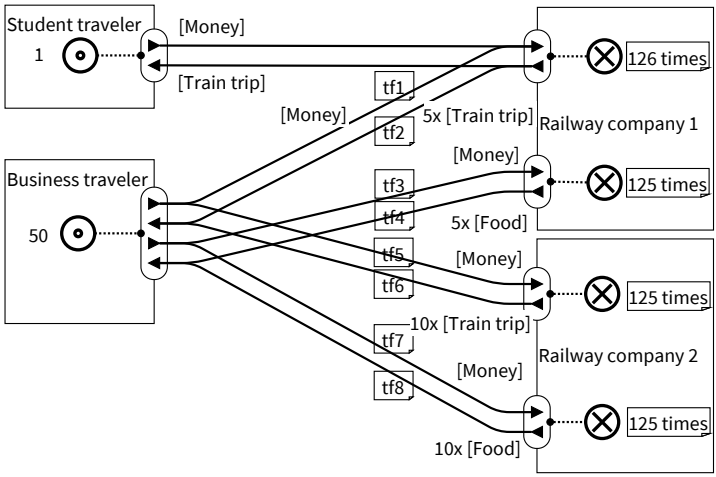

Figure 5.6: Cardinalities in a network with transaction choice and merge. The labels and numbers computed by the editor are shown in comments signs.

Table 5.1: Transaction choices of the business Traveler in figure 5.6.

Actor	*Tx*	*Ratio*	*Tf*	*Direction*
Business Traveler	tx1	1	tf1	to Railway company 1
			tf2	from Railway company 1
			tf3	to Railway company 1
			tf4	from Railway company 1
	tx2	1	tf5	to Railway company 2
			tf6	from Railway company 2
			tf7	to Railway company 2
			tf8	from Railway company 2
	tx3	0	tf1	to Railway company 1
			tf2	from Railway company 1
			tf7	to Railway company 2
			tf8	from Railway company 2
	tx4	0	tf3	to Railway company 2
			tf4	from Railway company 2
			tf5	to Railway company 1
			tf6	from Railway company 1

intended in figure 5.6 and in which ratio. The table shows that the intended transactions occur an equal number of times.

The numbers in a transaction table are ratios. Transaction txi is chosen

$$\frac{r_i}{\sum_j r_j}$$

times. In figure 5.6, transaction 2 is chosen 50% of the time, so the boundary elements in Company 2 are executed 125 times.

Figure 5.6 contains a transaction merge at the interface of Railway company 1. At a transaction merge, the numbers of the merged transactions are added, so the interface of Railway company 1 is triggered 126 times. These numbers are computed by the *e³value* analysis software and are shown as comments in figure 5.6.

Transaction tables specify the cardinality of transactions, and hence of transfers, which then override any cardinality specified at a port. For example, the company offers single Train trips to student Travelers, and indicates this with an explicitly specified port cardinality. This is the cardinality used in transactions with the student. In transaction with the business Traveler, the ratios specified for the transactions with the business Traveler are used.

The *e³value* tool contains a transaction analyzer, which in case of an ambiguous diagram like that in figure 5.6, generates a table of possible transactions and allows the user to select the ones that he or she intends. We explain this in detail in the next section.

5.7.1 Transaction dependency

Transaction dependencies have not yet been implemented.

In figure 5.7, A4 performs transaction tx2 = {t3, t4, t5, t6}with A1 and A2. However, A1 and A2 each can make a choice between tx2 and another transaction. This creates a **transaction dependency** between the choices of A1 and A2. Table 5.2 clarifies the situation.

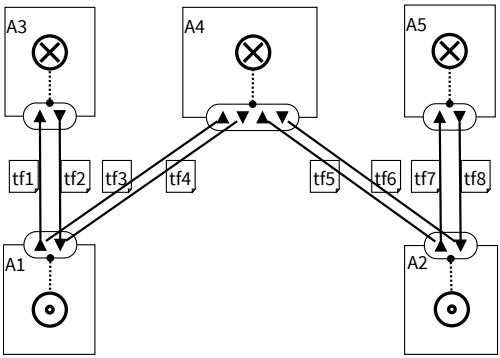

Figure 5.7: Dependencies enforced by an interface.

Table 5.2: Dependent transaction choices of A1 and A2 in figure 5.7.

Actor	*Tx*	*Ratio*	*Tf*	*Direction*
A1	tx1	8	t1	to A3
			t2	from A3
	tx2	2	t3	to A4
			t4	from A4
A2	tx2	3	t5	to A4
			t6	from A4
	tx3	4	t7	to A5
			t8	from A5

For the sake of the example, in table 5.2 A1 chooses between transactions tx1 and tx2 in an 8:2 ratio and A2 chooses between transactions tx2 and tx3 in a 3:4 ratio. The dependency between A1 and A2 is that A1 must choose tx2 in the contract period exactly the same number of times as A2 chooses tx2 in the same period.

Suppose A1 has 60 need occurrences. Then it executes tx1 48 times and tx2 12 times. Due to the transaction dependency, this means that A2 executes tx2 12 times too. Together with the choice ratio of A2 this implies that A2 has 28 need occurrences, split in absolute numbers over 12 times tx2 and 16 times tx3.

The single interface of A4 thus implies the transaction dependency that

$$\frac{r_2}{r_1 + r_2} * N_1 = \frac{r_3}{r_3 + r_4} * N_2.$$

where r_i is the ratio of transaction tx_i and N_j is the number of needs of Aj.

5.8 Valuations of a value transfer

Each value transfer has a monetary value for the actors involved in a transfer. For the transfer of money, this value is fixed: If I give you $f10$, then you receive $f10$. For the transfer of other value objects, the value of the object differs per actor.

All ports and transfers have a Valuation property that can be set. Its default value is undefined.

5.8.1 Money transfers

The amount of money transfered between actors is specified as Valuation at one of the ports or at the transfer itself. The Valuation of ports and value transfers is by default undefined. If an actor determines the price, we set the Valuation property at the port of that actor. During analysis the editor checks that the valuation of the port at the other end is undefined.

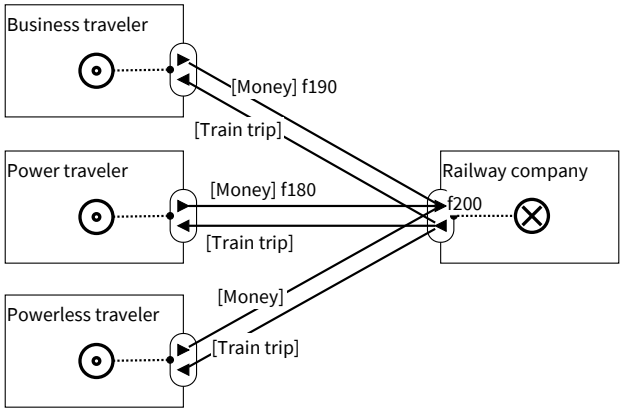

Figure 5.8: Two ways to specify how much is money is paid.

If the two actors agreed on a price, we set the Valuation property of the transfer. If the Valuation is specified for a transfer, this overrides any value specified at a port of the transfer.

For example, figure 5.8 shows a business Traveler who negotiated a price of $f190$ for a Train trip and a power Traveler forced the Railway company to accept a price of $f180$. The powerless Traveler pays the standard price of $f200$ set by the Railway company.

5.8.2 Non-money transfers

Non-money objects may have a different value for different actors. For example, in the market scenario in figure 5.9, the Traveler and the Railway company assign different values to non-money objects. For the Traveler, the valuation measures the increase in utility by consuming the value object, for the Railway company the valuation measures the cost of producing it.

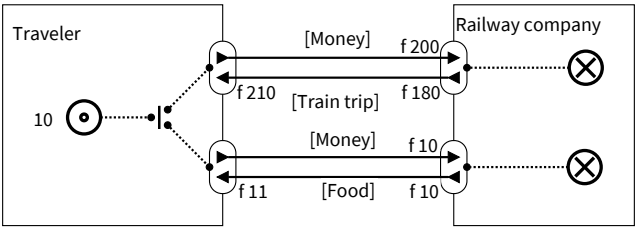

Figure 5.9: Adding valuation for non-money transfers.

5.9 Expenses

All money outflows in an *e³value* model are expenses, but not all expenses that an actor has need to be modeled as outflows. For example, companies have personnel expenses but we usually do not include a personnel market segment in the model that exchanges labor for salary with the company. Similarly, we usually do not show expenses on acquisition and maintenance of IT.

Nevertheless, these items may incur significant expenses that we want to include in an *e³value* model. We distinguish fixed expenses from variable expenses. **Fixed expenses** occur once in a market scenario and are associated with the actor, market segment or value activity that incurs the expense. **Variable expenses** depend on the volume of business and are associated with a port. These expenses are associated with producing or transferring a value object.

For example, in figure 5.10, the Railway company has fixed expenses of f70K in a market scenario and variable expenses of f34 per Train trip sale. It sells 500 000 Train trips, by which it has f34 * 500 000 = f17 000 000 variable expenses. Adding fixed expenses, total expenses in this market scenario are f17 070 000. It has a gross revenues of f200 * 500 000 = f100 000 000 and a net income of f100 000 000 - f17 070 000 = f82 930 000.

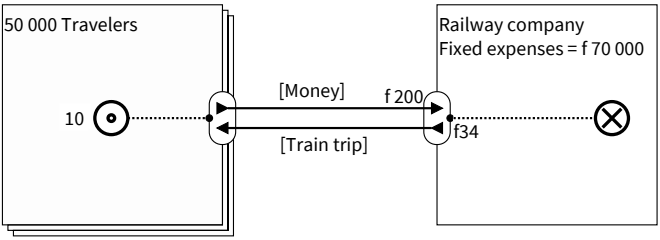

Figure 5.10: Expenses for the Travelers and Railway company. Expenses associated with a port are written inside the actor. A valuation associated with a port is written outside the actor.

5.10 Investments

To start a new business idea, investments are needed. Each actor may have an investment that will be subtracted from its revenue when computing its net cash flow in a market scenario (figure 5.11).

Investments are used in a discounted net present cash flow analysis in a sequence of market scenarios, called a time series (chapter 7). The difference with expenses is that expenses are incurred in every contract period whereas an investment is done only in the initial scenario of a time series.

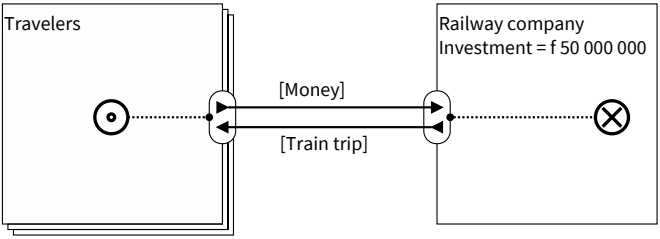

Figure 5.11: Upfront investments for Railway company.

Investment can also be specified for value activities and market segments. Its default value is 0.

5.11 Computing property values

All examples so far have used constant property values. However, in *e³value* , property values can be computed from other property values using an expression language that consists of two parts: a navigation language to refer to values defined in the model and a set of arithmetical operators to compute values. Appendix C defines the expression language of *e³value* . The *e³value* editor provides a simple technique by which you can instruct the editor to construct navigation expressions for you.

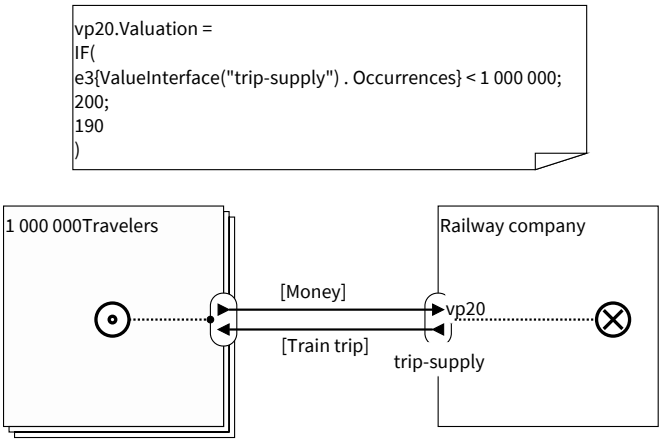

Figure 5.12: The pricing is dependent on the total number of Train trips sold.

Bulk price reduction. Consider the example in figure 5.12. In this model, the interface of the Railway company is called trip-supply. The valuation associated to the in-port of this interface is that the price of the trip is $f200$ if the number of occurrences in the market scenario is less than 1 000 000, otherwise it is $f190$.

In the example, the navigation expression evaluates to $f10,000,000 = 10 \times 1,000,000$ and so the train tickets in this market scenario cost $f190$

Variable price. In figure 5.13 the valuation expression of port vp12 computes the price of a Train trip as FixedTariff $+ 0.5 *$ Distance. FixedTarrif is a user-defined property of the port vp12 and Distance is a user-defined property of the value transfer ve17.

```
vp20.Valuation =
e3{Model() . Actor("Railway company") . ValueInterface("vi13") . in-offering() .
ValuePort("vp20") . FixedTarrif}
+
0.5 * e3{Model() . Actor("Traveler") . ValueInterface("vi7") . in-offering() . ValuePort("vp6") .
upstream-transfer("tf17") . Distance}

vp20 . FixedTarrif = 3
```

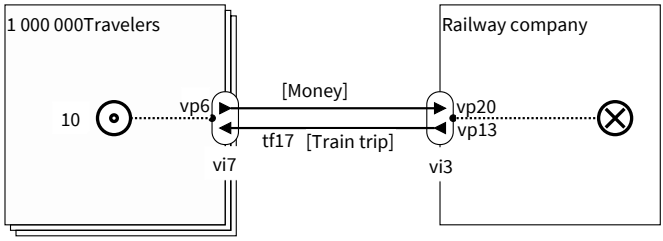

Figure 5.13: A Traveler obtains a Train trip from a Railway company.

Chapter 6

Net Value Flow Analysis

This chapter describes the quantification capabilities of the *e³value* tool. This has not yet been implemented in the *e³web* tool.

In net value flow analysis of a market scenario we follow a trace through the model from customer needs to boundary elements, and compute the value flows into and out of the actors along this trace. for each actor, these numbers are added to give its *net value flow*. If the net value flow of an actor is positive, the scenario is financially sustainable for this actor. If the net value flow for all actors in a value network is positive, then the scenario is financially sustainable for the entire value network. If we do a value flow analysis for the money transfers only, we get a *cash flow analysis.*

First we define traces and then we explain the net value flow analyses for actors, market segments, value activities and partnerships.

6.1 Traces

Dependency paths of a value model may overlap, as illustrated in figure 6.1. This models the situation that the Railwaycompany offers group tickets and two Travelers use this offer.

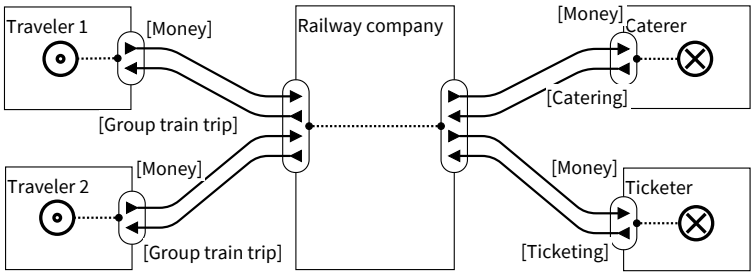

Figure 6.1: A value model with two overlapping dependency paths.

In general, a value model has one or more customer needs, each at the root of a dependency graph, and these graphs may or may not overlap. A **trace through the model** is a set of dependency paths, one for each customer need in the model. If the dependency graphs of the model are disjoint, then we can partition a trace in disjoint dependency paths and analyze them separately. But if the dependency graphs overlap, then a trace will contain partially merged dependency paths that must be analyzed jointly.

6.2 Net value flow analysis of an actor

To analyze a market scenario, we start from the needs in the scenario and trace these to their boundary elements. To do this, we quantify the model and then let the *e³value* tool compute net value flow sheets of the scenario for each actor. A **net value flow sheet** is a spreadsheet that lists the inflows and outflows of an actor and

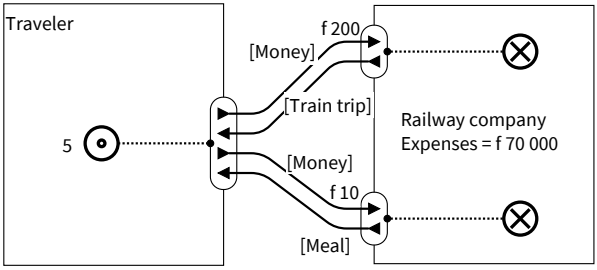

Figure 6.2: Traveling from Amsterdam to Paris.

sums them into the net value flow of the actor. For example, for the scenario in figure 6.2 the e^3value tool can compute the net value flow sheets shown in tables 6.1 and 6.2.

In a net value flow sheet we identify an interface by its unique name or by the value objects entering and leaving the interface, if that is unique for the actor.

The sheet shows the number of occurrences of the interface in the market scenario. For each port of the interface, the sheet lists the type of value object and identifies the transfers being counted. In case of transaction choice or merge, more than one transfer is connected to a port. The table then shows for each transfer the number of occurrences of the transfer in this market scenario, the valuation of the transfer, and the total value of the occurrences being counted.

Next, the net value passing through each interface is summed, expenses are subtracted, and this gives the net value flow for each actor. In table 6.1, the Railway company has a net value flow of f-68 950. This shows that with fixed expenses of f70K and only one traveler, the Railway company cannot sustain itself. With a market segment of sufficient size (section 6.3), Railwaycompanies can generate positive revenue.

The Traveler ends up with a total value of f-1 050 because we only

Table 6.1: Net value flow sheet of the Railway company of figure 6.2.

Interface	Port	Transfers	Transfer occurrences	Valuation	Total value transferred	Net value flow
Meal, Money			5			50
	out: Meal	all	5	0	0	
	in: Money	all	5	10	50	
Money, train trip			5			1K
	in: Money	all	5	200	1 K	
	out: Train trip	all	5	0	0	
Expenses						-70 K
Total for actor						-68.95 K

Table 6.2: Net value flow sheet of the Traveler of figure 6.2.

Interface	Port	Transfers	Transfer occurrences	Valuation	Total value transferred	Net value flow
Train trip, Meal, Money, Money			5			-1050
	in: Train trip	all	5	0	0	
	in: Meal	all	5	0	0	
	out: Money	all	5	10	-50	
	out: Money	all	5	200	-1K	
Total for actor						-1050

show the cash flows in the sheet. The traveler spends this amount of money on train trips and so the traveler *cash flow* is negative. Assuming that the value of a train trip and food is higher than the $f210$ that the traveler pays for it, the *net value flow* for the traveler is positive. We will include this valuation in the example of section 6.3.

6.3 Net value flow analysis for a market segment

Figure 6.3 shows a value network with two market segments. The numbers given for a market segment are averages for the actors in the segment. There are 1 000 000 Travelers with on the average each 10 needs for a train trip. There are 2 Railway companies which sell on the average 10 000 000 / 2 = 5 000 000 train trips.

Tables 6.3 and 6.4 show the net value flow sheets for individual actors in the market segments. Each Railway company has on the average $f70$K fixed expenses in the market scenario and $f34$ per sale of a trip. Plugging in these numbers gives is a net value flow of $f57$ 930 000 per Railway company. With a market of this size, providing train trips is profitable.

Travelers valuate a train trip on the average at $f210$ and the average trip price for the two Railway companies is $f200$. The net value flow for Travelers is on the average $f100$.

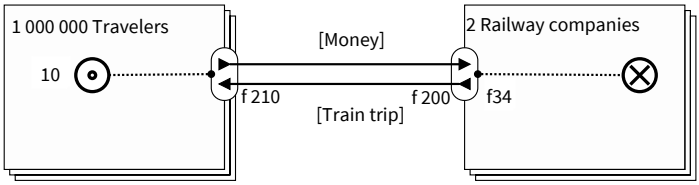

Figure 6.3: A Traveler and a Railway company as market segments.

Table 6.3: Net value flow sheet of the Railway companies market segment of figure 6.3.

Interface	Port	Transfers	Transfer occurrences	Valuation	Total value transferred	Net value flow
Money, Train trip			5 M			58 M
	in: Money	all	5 M	15	75 M	
	out: Train trip	all	5 M	0	0	
	out: Train trip	Expenses	5 M	34	-17 M	
Expenses						-70 K
Total for actor						57.93 M
Market segment size						2

Table 6.4: Net value flow sheet of the Travelers market segment of figure 6.3.

Interface	Port	Transfers	Transfer occurrences	Valuation	Total value transferred	Net value flow
Train trip, Money			10			100
	in: Train trip	all	10	210	2100	
	out: Money	all	10	200	-2000	
Total for actor						100
Market segment size						1 M

6.4 Net value flow analysis of a value activity

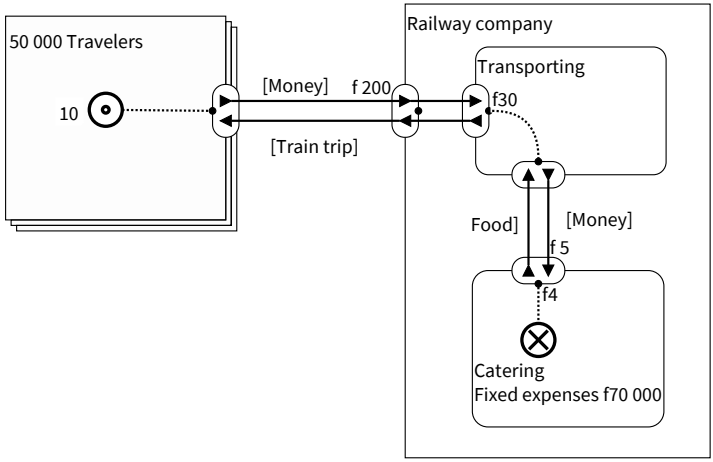

Figure 6.4: Quantified value activities.

Figure 6.4 shows two value activities of the Railway company. We can generate value flow sheets for each activity to check if they each have a positive net value flow in a market scenario. This is important to make the decision whether or not to outsource the activity. There are two cases.

(1) If the net value flow of a value activity is positive, it is profitable to continue it. It may be outsourced, but then the outsourcing party must be able to do it cheaper than the internal activity does it.

(2) If the net value flow of a value activity is negative, then it is not a profit center and should be removed from the model. The cost of the activity may appear as an increased fixed expense.

Alternatively, it may be moved to another actor, who may be able to perform the activity in a profitable way. The price for the activity should be lower than the fixed expenses incurred if done in-house.

To generate a value flow sheet for a value activity, we must find the valuations of the value activity. We do this by distinguishing **reciprocal transfers** from **forwarding transfers.** A reciprocal transfer is part of a transaction and connects ports with opposite roles, i.e. it connects an out-port and an in-port. A forwarding transfer connects ports with the same role.

The interface of a value activity may be connected by a forwarding transfer to the interface of a higher-level entity (a market segment, actor of higher-level value activity). In that case the interface of the value activity inherits its valuations from this higher-level entity.

Alternatively, the interface of a value activity may be connected by a reciprocal transfer to the interface of another value activity. In that case the valuation must be specified as discussed above for actor interfaces. An example will make this clear.

In figure 6.4, the Transporting activity is connected by forwarding transfers to the interface of the Railway company. It inherits its valuations from those of the Railway company. So the Transporting activity receives $f200$ for providing a Train trip.

Transporting is connected to Catering by reciprocal transfers. So it pays $f5$ for Food to the Catering activity.

Transporting incurs $f30$ variable expense for providing a Tain trip, and Catering incurs $f4$ for providing Food.

Tables 6.6 and 6.7 show the net value flow sheets of the activities, including fixed and variable expenses. The Travelers market segment consists of 50 000 passengers who on the average make 10 trips each for $f200$, so the interface of the Railway company is triggered 500 000 times and the total value flowing in is $f100$M. By implication, this is also the total value flowing through the corresponding interface into the Transporting activity. The variable expense to generate this inflow is 15M, so the net value flow across this activity interface is $f85$M.

Following the dependency path, we derive that the catering interface is triggered 500K times and the total value flowing into that activity is $f2.5$M. Fixed expenses are high, and the margin of $f430$K is rather small compared to that of transporting.

Table 6.5: Net value flow sheet of the Railway company of figure 6.4 with its value activities consolidated.

Interface	Port	Transfers	Transfer occurrences	Valuation	Total value transferred	Net value flow
Money, Train trip			500 K			100 M
	in: Money	all	500 K	200.00	100 M	
	out: Train trip	all	500 K	0.00	0.00	
Fixed expenses						-17.07 M
Total for actor						82.93 M

Table 6.5 shows the **consolidated net value flow sheet** of the Railway company. The fixed expenses in the net value flow sheet of the Railway company consist of

- the fixed expenses of the Railway company,
- the fixed expenses of all value activities, and
- the variable expenses of all value activities.

Table 6.6: Net value flow sheet of the Transporting activity of figure 6.4.

Interface	Port	Transfers	Transfer occurrences	Valuation	Total value transferred	Net value flow
Money, Train trip			500 K			85 M
	in: Money	all	500 K	200.00	100 M	
	out: Train trip	all	500 K	0.00	0.00	
	out: Train trip	Variable expenses	500 K	30.00	-15 M	
Catering, Money			500 K			-2.5 M
	in: Catering	all	500 K	0.00	0.00	
	out: Money	all	500 K	5.00	-2.5 M	
Total for actor						82.5 M

Table 6.7: Net value flow sheet of the Catering activity of figure 6.4.

Interface	Port	Transfers	Transfer occurrences	Valuation	Total value transferred	Net value flow
Catering, Money			500 K			500 K
	out: Catering	all	500 K	0.00	0.00	
	out: Catering	Variable expenses	500 K	4.00	-2 M	
	in: Money	all	500 K	5.00	2.5 M	
Fixed expenses						-70 K
Total for actor						430 K

6.5 Net value flow analysis of a partnership

The net value flow sheet of a partnership consolidates that of the participating actors, except that the internal expenses of the actors are not included in the partnership value flow. Figure 6.5 shows a partnership of a Railway company and a Caterer. Table 6.8 shows the consolidated value flow sheet for the partnership and table 6.9 shows that the caterer has expenses not included in the value flows of the partnership.

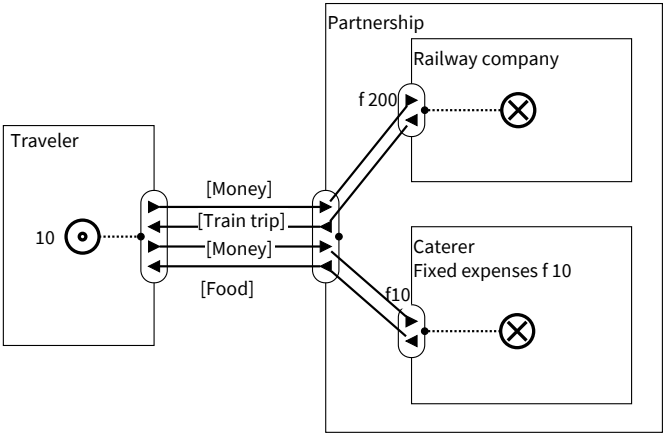

Figure 6.5: The Railway company and the Caterer form a partnership to offer a train trip and food as a bundle.

Table 6.8: Net value flow sheet of the partnership of figure 6.5.

Interface	Port	Transfers	Transfer occurrences	Valuation	Total value transferred	Net value flow
Money, Money, train trip, food			10			2100
	in: Money	all	10	10	100	
	in: Money	all	10	200	2000	
	out: train trip	all	10	0	0	
	out: food	all	10	0	0	
Total for actor						2100

Table 6.9: Net value flow sheet of the Caterer of figure 6.5.

Interface	Port	Transfers	Transfer occurrences	Valuation	Total value transferred	Net value flow
Money, food			10			50
	in: Money	all	10	10	100	
	out: food	all	10	0	0	
	out: food	Variable expenses	10	5	-50	
Total for actor						50

6.5.1 Independent valuations

An actor may participate in a partnership and at the same time do independent business. This may result in a port connected with reciprocal and forwarding transfers at the same time.

In figure 6.6, the interface of the Railway company is connected to the partnership interface with a forwarding transfer and to the Corporate traveler through reciprocal transfers. In transactions through the partnership interface it will use the valuations of that interface but in transactions witgh the Corporate traveler it will use the price set for or agreed with the Corporate traveler.

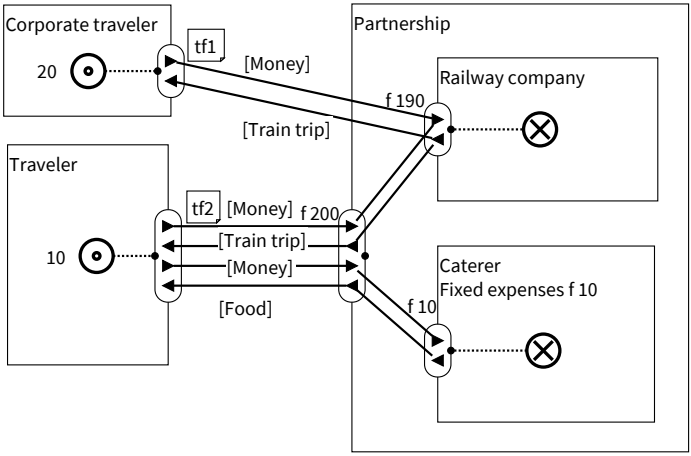

Figure 6.6: The Railway company and the Caterer form a partnership to offer a train trip and food as a bundle.

Table 6.10: Net value flow sheet of the Railway company of figure 6.6.

Interface	**Port**	**Transfers**	**Transfer occurrences**	**Valuation**	**Total value transferred**	**Net value flow**
Money, train trip			30			5.8 K
	in: Money	tf1	20	190	3.8 K	
	in: Money	tf2	10	200	2 K	
	out: train trip	all	30	0	0	
Total for actor						5.8 K

Chapter 7

Discounted Net Value Flow Analysis of a Time Series

This chapter describes the quantification capabilities of the *e³value* tool. This has not yet been implemented in the *e³web* tool.

New business ideas require investment and usually lead to a negative net cash flow initially. To analyze the financial sustainability of the idea, we need to do a net cash flow analysis of a *sequence* of market scenarios, which we call a *time series.* There are several methods to compute return on investment, including calculation of the payback period of investments, internal investment rate calculation, real option theory, and Discounted Net Present Cash Flow (DNPC) technique [3]. In this chapter we show how to do a discounted net value flow computation of a time series.

7.1 Time series

A **time series** is a sequence of market scenarios for consecutive contract periods. Each market scenario quantifies an e^3value model. In an investment analysis, the models in a time series usually are the same and the only thing that is different is the quantification in the consecutive market scenarios. However, for the computations that follow that is not important and we may create a time series where consecutive models are different.

7.2 Naive net present cash flow analysis

For one contract period, we sum all all in-going cash flows of an actor over the contract period and subtract the sum of all out-going cash flows, including investments. If we do this for all periods in a time series, the net present cash flow of a time series is

$$\sum_{p=0}^{m}(Revenues_p - Expenses_p - Investments_p)$$

where p ranges over the periods of the time series.

For example, consider table 7.1. In period 0, an investment is made and no revenues and expenses occur yet. In periods 1, 2 and 3 we have a revenues of $f500$ and expenses of $f100$ in each period. This gives a positive net cash flow of $f200$ at the end of the time series.

7.3 Discounted net present cash flow analysis

To assess a possible investment, we must take the time value of money into account [3]. We could invest the $f1{,}000$ in risk-free state-bonds, which are guaranteed to pay out a certain interest over

Table 7.1: Naive calculation of net cash value over a series of market scenarios.

Period	Revenues	Expenses	Invest-ments	Total
0			1,000	-1,000
1	500	100		400
2	500	100		400
3	500	100		400
Total				200

the investment. Suppose that this interest is 5%. The an investment of $f1{,}000$ in state bonds will grow to $f1{,}000 \times 1.05 = f1{,}050$ after one year.

Reasoning the other way around, we will have $f1{,}000$ in one year if we invest $f952.38$ now in risk-free state bonds at 5% interest. We say that the net present value of $f1{,}000$ next year is $f952.38$ now, or that $f1{,}000$ next year is discounted to $f952{,}38$ now.

In general, the possession of fX n years from now, is worth fY now, if we can make Y grow into X according to the best, risk-free opportunity available to us now. Discounting a future amount X to its current value Y incorporates our estimate of what is the best risk-free opportunity available to us now.

As a consequence, the *same* amounts of money in *different* future contract periods have a different discounted value today.

In realistic situations, many consecutive contract periods are needed to earn back an investment. To assess the value of an investment and compare it to a risk-free investment is state bonds, all money values in these consecutive periods should be discounted to the same time period. Usually this is period 0, the time at which the investment was made.

To calculate the net discounted present cash for a series of time periods, the following formula for **Discounted Net Present Cash**

(DNPC) Flow calculation can be used:

$$\sum_{p=0}^{m} \frac{(Revenues_p - Expenses_p - Investments_p)}{(1 + interest)^p}$$

where p ranges over the time periods of the time series.

Using this formula for the investments, revenues, and expenses in table 7.1, using an interest rate of 5 %, results in table 7.2. The net revenue of $f400$ generated in each period is worth less the farther the period lies in the future. The value of our investment is now only $f89.30$.

Table 7.2: Calculation of discounted net present cash value over a series of market scenarios using 5% interest

Period	Revenues	Expenses	Invest-ments	DNPC
0			1,000	-1,000
1	500	100		380.85
2	500	100		362.81
3	500	100		345.54
Total				89.30

7.4 The cost of risk

Suppose now we want to invest in a new venture, with considerably more risk than investing in state bonds. This increased risk is represented by a higher interest rate. In addition, the interest rate is also affected by *cost of capital*, since most companies must do some work to find the money to do the investment. High risk and high cost of capital are reflected in a high interest rate used for discounted net present value computations.

Suppose we value our risk at 20%. Using this percentage, the net cash flows generated a few years from now discount to a smaller net present value, as shown in table 7.3. With this amount of risk, the investment seems not to be such a good idea, since the investment results in a *negative* DNPC. For an investment with risk valued at 20% to be profitable, yearly net revenue should have been much greater.

Table 7.3: Calculation of discounted net present cash value over a series of time periods using 20% interest

Period	Revenues	Expenses	Invest-ments	DNPC
0			1,000	-1,000
1	500	100		333.33
2	500	100		277.78
3	500	100		231.48
Total				-157.41

A frequent mistake while calculating the DNPC is to include depreciation costs [3]. However, depreciation costs do not lead to direct expenses. Rather, the upfront investment for which depreciation cost are made has already been taking into account while doing a DNPC calculation, so considering depreciation costs also, would lead to double counting.

Finally, it is important to understand a few assumptions while calculating net present discounted cash flows. First, the cash (revenues and expenses) should *directly* relate to the investment made.

Second, the calculation supposes that positive results of a contract period in a time series can be invested with same interest rate, as was used for the calculation.

Third, the calculation supposes that cash flows are only occurring at specific points in time, namely at the end of each contract period. Obviously, that is not always the case in real-life, but a simplification

is made here to allow for easy calculation.

7.5 Discounted net present cash flow computation in the time series tool

Figure 7.1 gives an example of a time series.

Period 0 is the time of investment and we have 0 sales. This leads to a net value flow sheet for period 0 as shown in table 7.4.

In period 1 and later, there are no investments. In period 1 there are 50,000 Travelers, who have a need to travel 10 times. For each trip, they pay $f200$.-. The net value sheet for period 1 is shown in table 7.5.

In period 2, the Railway company attracts more customers, so the market segment Travelers now has 60,000 actors rather than just 50,000. This results in a changed net value sheet for period 2 (table 7.6).

In period 3, the Travelers decide to travel 12 times and not just 10 times (table 7.7).

In period 4, the Railway company has increased the price of a train trip $f210$. Everything else staying the same in this time series, this results in a net value flow increase for the Railway company as shown in table 7.8.

The net value flow sheets for the periods of the time series in figure 7.1 are shown in tables 7.4 to 7.8.

Each time series as a whole has an **interest rate** that will be used in DNVF computations. Each actor and market segment can have an interest rate too, in which case the DNVF algorithm takes the interest rate for that actor or market segment.

Using an interest rate of 7%, the discounted net present cash flow of the Railway company is $f381{,}167{,}241.65$. If this is higher than the discounted net present value of investing $f50$M in risk-free state bonds, then the railway company is worth an investment of $f50$M. If it is *not* higher, then this means that the activities of the Railway

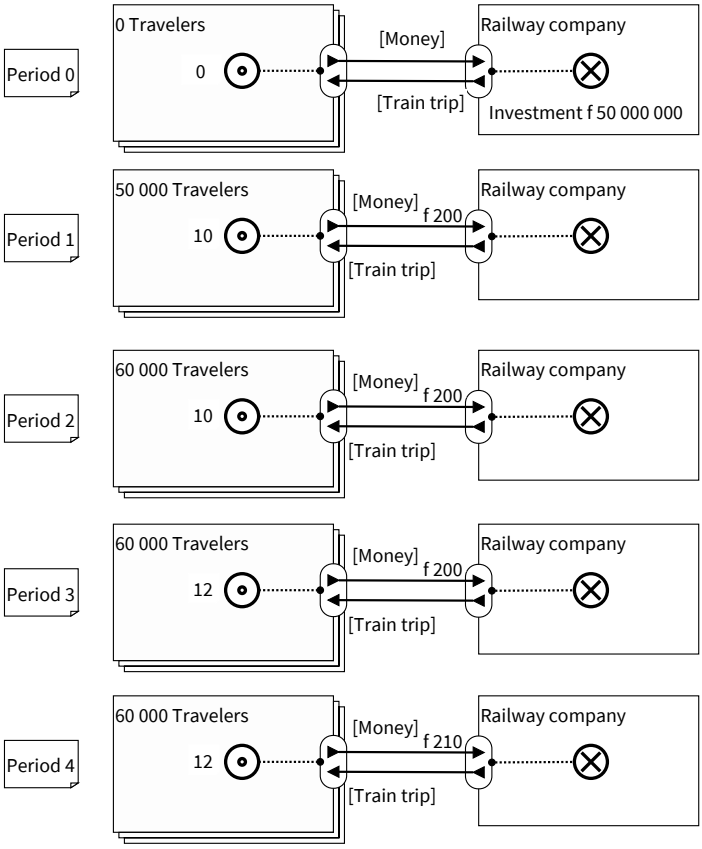

Figure 7.1: A time series for a Railway company.

company do not add any value with respect to an investment in risk-free state bonds.

Table 7.4: Net value flow sheet of period 0 of the Railway company of figure 7.1.

Interface	Port	Transfers	Transfer occurrences	Valuation	Total value transferred	Net value flow
traintrip, Money			0.00			0.00
	out: traintrip	all	0.00	0.00	0.00	
	in: Money	all	0.00	200.00	0.00	
Investment					50 M	
total for actor						-50 M

Table 7.5: Net value flow sheet of period 1 of the Railway company of figure 7.1.

Interface	Port	Transfers	Transfer occurrences	Valuation	Total value transferred	Net value flow
rain trip, Money			500 K			100 M
	out: Train trip	all	500 K	0.00	0.00	
	in: Money	all	500 K	200.00	100 M	
total for actor						100 M

Table 7.6: Net value flow sheet of period 2 of the Railway company of figure 7.1.

Interface	Port	Transfers	Transfer occurrences	Valuation	Total value transferred	Net value flow
Money, Train trip			600 K			120 M
	in: Money	all	600 K	200.00	120 M	
	out: traintrip	all	600 K	0.00	0.00	
total for actor						120 M

Table 7.7: Net value flow sheet of period 3 of the Railway company of figure 7.1.

Interface	Port	Transfers	Transfer occurrences	Valuation	Total value transferred	Net value flow
Train trip, Money			720 K			144 M
	out: Train trip	all	720 K	0.00	0.00	
	in: Money	all	720 K	200.00	144 M	
total for actor						144 M

Table 7.8: Net value flow sheet of period 4 of the Railway company of figure 7.1.

Interface	Port	Transfers	Transfer occurrences	Valuation	Total value transferred	Net value flow
Money, Train trip			720 K			151.2 M
	in: Money	all	720 K	210.00	151.2 M	
	out: Train trip	all	720 K	0.00	0.00	
Total for actor						151.2 M

Chapter 8

Fraud Scenarios

This chapter describes the quantification capabilities of the e^3tool . This has not yet been implemented in the e^3web tool.

The market scenarios of a value model assume that all actors perform their transactions as specified. A **fraud scenario** is a market scenario in which some actors do not behave as agreed in a value model, for example by not paying or by engaging in a secret transaction. In this chapter we describe the structure of fraud scenarios and explain how they can be analyzed.

We use the the value model of figure 8.1 as example. We assume that User A has a flat rate subscription with Provider A and User B has a monthly subscription with Provider B. If User A calls User B, there will be an interconnection between the two providers.

8.1 Representing fraud scenarios

In a fraud scenario, we mark some actors as trusted. All actors have agreed on the value model, but non-trusted actors may do things that violate the value model. Trusted actors always conform

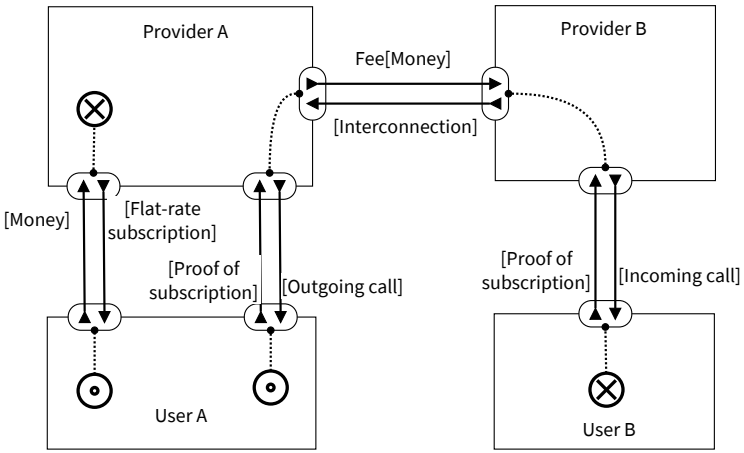

Figure 8.1: A value model of telecom subscriptions. We assume that User B has a monthly subscription with Provider B.

to the value model.

Given a selection of trusted actors, a fraud scenario contains three elements not present in a normal market scenario (figure 8.2):

- *Absent transfers,* represented by a dashed transfer. For example, a non-trusted actor may not pay for a value object received.
- *Hidden transfers,* represented by a dotted transfer. For example, non-trusted actors may engage in a value transfer that is not in the value model. The trusted actors are not aware of these transfers.
- *Collusions,* represented by a fat border. Two or more actors may collude, meaning that there revenue flows are added and then split according to a formula that is advantageous for all of them.

In the fraud scenario of figure 8.2, the three fraudulent activities are present.. In this scenario, Provider A is trusted. The model has been quantified in such a way that User B now has a revenue-sharing subscription with Provider B, in which User B receives a small share of the revenue that Provider B gets from the interconnection fee. This is unknown to Provider A. User B colludes with User A by by sharing the revenue that it receives from Provider B.

With a flat-rate subscription, User A can call people for a flat rate per month, within a fair use policy (e.g. at most 500 calls per month). In our scenario, User A calls User B the maximum number of times within its fair use restriction and colludes with User B in sharing the revenue that User B receives from Provider B. On top of that, User A does not pay Provider A for the flat rate subscription.

8.2 Analyzing fraud scenarios

E³tool contains functionality to generate all fraud scenarios and their cash flows from a quantified value model. The tool can order these

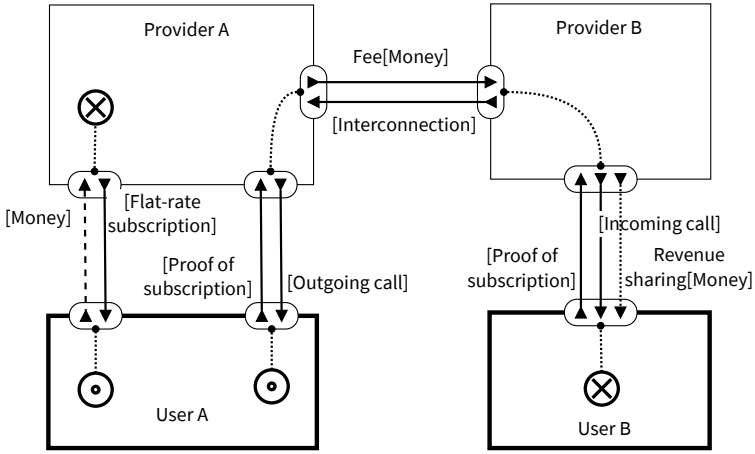

Figure 8.2: A fraud scenario for the flat rate telecom model. Provider A is trusted. Invisible for Provider A, User B has a revenue-sharing subscription with Provider B.

scenarios on cost for the victim or net revenue for the attackers. Cost for the victim is a measure of the impact of the fraud on the victim. Revenue for the attackers is a mesure for the likelihood that the attck will take place.

Alternatively, you can create a fraud scenario yourself and generate the cash flows with *e*^{3}*tool* .

Appendix A

The *e*^{3}*value* Ontology

In the ontology diagrams, the role that a concept is playing in a relationship is written next to the concept, and the cardinality of the concept in the relationship is written at the far end. The default cardinality of the relationships is any (0..*). For example, in figure A.1, a **market segment** *performs* any number (0..*) of **value activities**, and a **value activity** is *performed-by-ms* at most one (0..1) **market segment**.

For each concept, its properties are listed (see appendix B for definitions). If a property has a default value, it is shown. The **name** property of concepts is initialized with a unique value by the *e*^{3}*value* editor.

If its value is derived from other property values, then it is preceded by an asterisk.

The *e*^{3}*value* editor allows you to add your own properties to any concept.

In figure A.1, value activities have an exclusive relationship either with a market segment or with an actor. This is true even if an actor is a member of a marlet segment.

Partnership only inherits the name of Actor. In *e*^{3}*value* we cannot specify investments, fixed expenses or interest of a partnership.

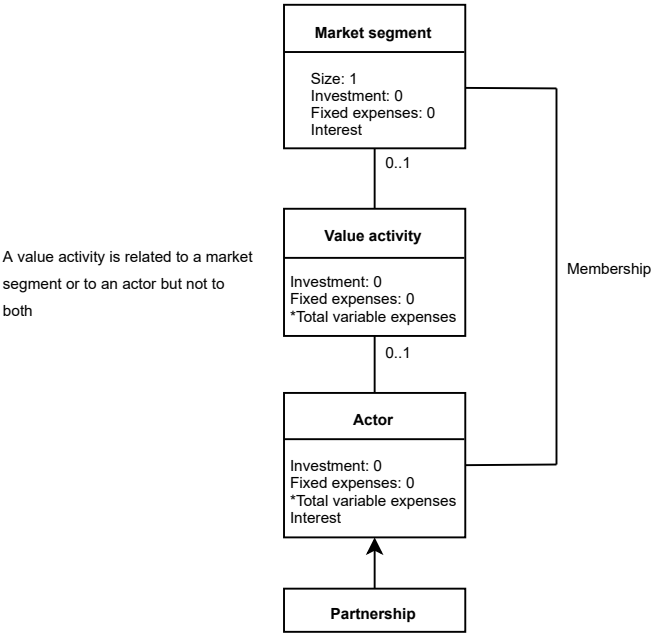

Figure A.1: Value network concepts (chapter 2). The arrow represents generalization.

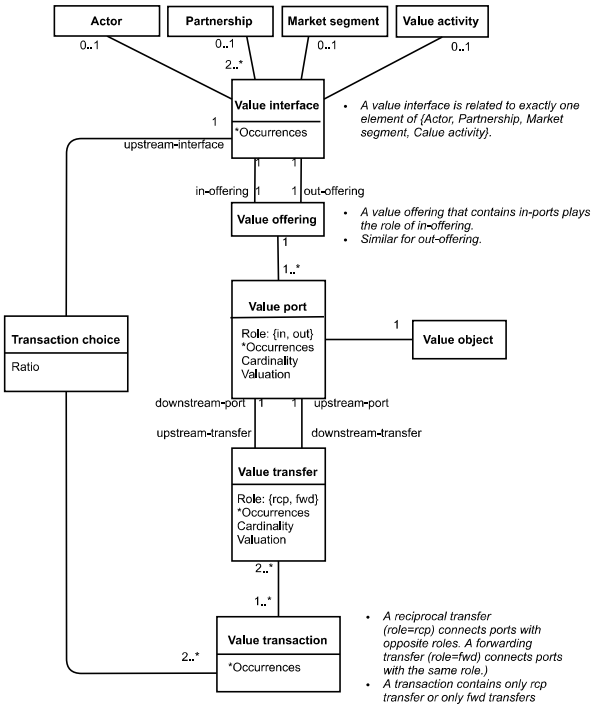

Figure A.2: Economic transactions (chapter 3).

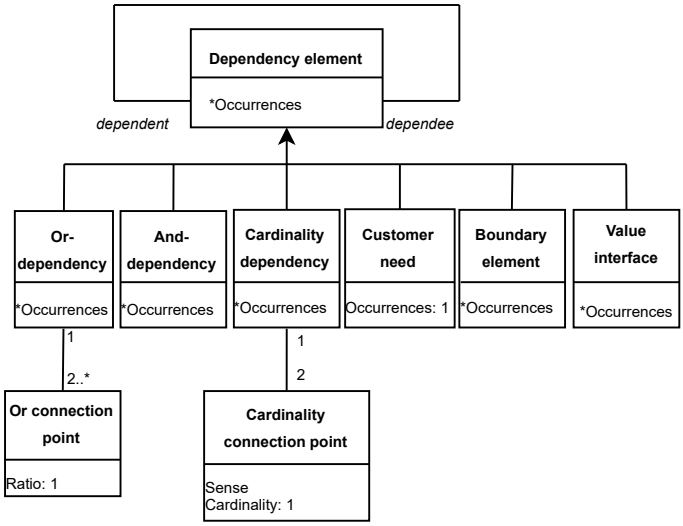

Figure A.3: Dependency elements (chapter 4). The arrow represent generalization.

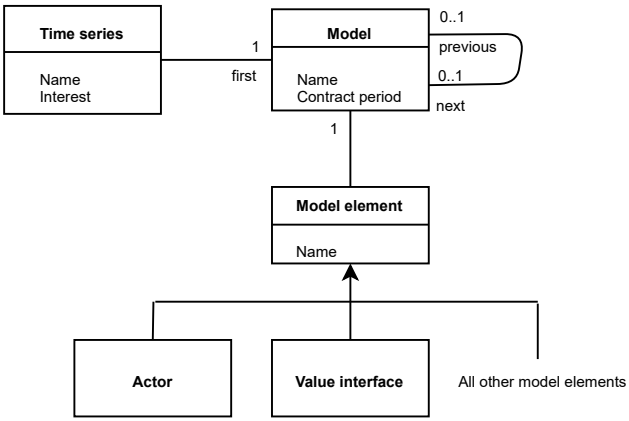

Figure A.4: Time series (chapter 7). The arrow represents generalization.

Appendix B

Properties

Model elements have properties that are used to count occurrences, valuate objects and estimate net revenue flow in a market scenario. The properties listed here are part of the ontology. For any model element, the user can define additional properties in the *e*^{3}*value* editor.

Most properties have a default value, listed below. If no default value is listed, this has a specific meaning in the execution of a market scenario, as explained below.

Property values can be set by a constant or computed by an expression (appendix C).

Actor

An entity that is responsible for its survival and well-being.

- **Name.** Unique name of the actor.
- **Fixed expenses.** Fixed expenses of an actor in a market scenario. Default value 0.

- **Total variable expenses.** Sum of the variable expenses at the ports of the actor. Derived value.
- **Investment.** The initial investment of an actor in a time series. Default value 0.
- **Interest.** If defined, the interest rate used for this actor in a discounted net present cash flow computation. Default value is undefined.

And-dependency

A logical conjunction of dependencies.

- **Name.** Unique name of the and-dependency.
- **Occurrences.** Positive integer indicating the number of times that an and-dependency is triggered in a market scenario. Derived value.

Boundary element

Represents the limit of the scope of a value model.

- **Name.** Unique name of the boundary element.
- **Occurrences.** Positive integer indicating the number of times that a boundary element is triggered in a market scenario. Derived value.

Customer need

A lack of something valueable that the actor wants to acquire.

- **Name.** Unique name of the customer need.

- **Occurrences.** Positive integer indicating the number of times that a customer need occurs in a market scenario. Default value is 1.

Cardinality dependency

A jump, up or down, in cardinality along a dependency path.

- **Name.** Unique name of the cardinality dependency.
- **Occurrences.** Positive integer indicating the number of times that the dependency is triggered in the current market scenario. Derived property.

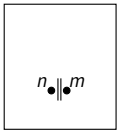

Figure B.1: A cardinality dependency.

Cardinality connection point

One side of a cardinality jump.

- **Sense.** Upstream or Downstream, depending on its position in a dependency path. Derived value.
- **Cardinality.** Positive integer indicating the relative number of times that one connection point occurs in the current contract period per occurrences of the other connection point of the cardinality dependency. If the upstream point in a dependency path has cardinality n and the downstream point has

cardinality m then for k occurrences of the upstream path, there are $k(m/n)$ downstream occurrences (figure B.1). If this is not a natural number, the scenario is inconsistent.

Default value is 1.

Market segment

A set of actors that assign value to objects in the same way.

- **Name.** Unique name of the market segment.
- **Size.** Positive integer indicating the number of actors in the market segment in a market scenario. Default value is 1.
- **Fixed expenses.** Average fixed expenses per actor in the market segment in the market scenario. Default value is 0.
- **Investment.** The initial investment of each actor in the market segment in a time series. Pre-defined property. Default value 0.
- **Interest.** If defined, the average interest the actors in the market segment use to computed discounted net present value of a time series. Default value is undefined.

Or-dependency

- **Name.** Unique name of the or-dependency.
- **Occurrences.** Positive integer indicating the number of times the or-dependency is triggered in a market scenario. Derived value.

Or connection point

A logical disjunction of dependencies.

- **Ratio.** Positive integer used to compute the number of occurrences of a disjunct in a market scenario. If disjunct i has ratio m_i, then n occurrences of the or-dependency in the contract period correspond to $n * (m_i / \sum_j m_j)$ occurrences of point i in the contract period. For example, in figure B.2, there are n occurrences of the or-dependency in the contract period exactly if there are $n * (m_i/(m_1 + m_2 + m_3))$ of disjunct m_i. Default value is 1.

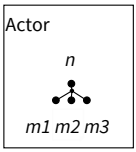

Figure B.2: Each disjunct has a ratio number.

Transaction choice

Choice of transactions departing from the same ports.

- **Name.** Unique name of the transaction choice.
- **Ratio.** Natural number that indicates the relative number of occurrences in a choice transaction. Default value is 1.

Value activity

A task performed by an actor that potentially results in a benefit for the actor.

- **Name.** Unique name of the value activity.
- **Total variable expenses.** Derived value.
- **Investment.** The investment done by this value activity at the start of a time series. Default value is 0.
- **Expenses.** Fixed expenses of the value activity. Default value is 0.

Value interface

A set of incoming and outgoing value ports of an actor that forms an atomic value exchange.

- **Name.** Unique name of the value interface.
- **Occurrences.** Positive integer indicating the number of times a value interface is triggered in a market scenario. Derived value.

Value offering

The collection of ports with the same role in a value interface.

- **Name.** Unique name of the value offering.

Value port

A willingness to provide or request value objects.

- **Name.** Unique name of the value port.
- **Role.** In or out. Derived value.
- **Occurrences.** Positive integer indicating the number of times a value port is triggered in a market scenario. Derived value.

- **Cardinality.** Positive integer that indicates the number of copies of a value object that is transferred when the port is triggered. Default value is undefined. During model analysis it is checked that the cardinality of at most one port of each value transfer is defined.

- **Valuation.** The value assigned by the actor to the value object passing through this port. Default value is undefined.

  If you set the valuation of a port in a diagram, it must be connected to a transfer. If this is a *money transfer,* then the valuation of the port at the other end of the value transfer must be undefined. If it is a *non-money* transfer, then both ports may have a valuation and these valuations may be different.

- **Expenses.** Variable expenses made in one occurrence of the port. Default is 0.

Value transfer

A willingness of a provider and a requester to transfer a value object from the first to the second.

- **Name.** Unique name of the value transfer.

- **Occurrences.** Positive integer indicating the number of times a value transfer is triggered in a market scenario. Derived value.

- **Cardinality.** Positive integer indicating the number of instances of a value object that is transferred when the transfer is triggered. By default the cardinality is undefined. If the transfer cardinality is defined, it overrides the cardinality of the ports to which it is connected.

- **Valuation.** Value of the transferred value object agreed by the provider and requester. By default the valuation is undefined.

If the valuation is defined, it overrides the value of the ports to which it is connected.

Value transaction

The set of value transfers trigered when a value interface is triggered.

- **Name.** Unique name of the value transaction.
- **Occurrences.** Positive integer indicating the number of times a transaction is triggered in a market scenario. Derived value.

Appendix C

The *e³value* Expression Language

E³value properties can be computed by expressions that refer to other properties in the same model. The language to do this consists of two parts, one to construct numerical expressions and one to construct navigation paths.

To do value flow analysis, the editor transforms the navigation path into an Excel cell reference and the resulting expression is evaluated in Excel. Hence, any Excel formula can be used as an *e³value* expression. See https://www.excelfunctions.net/excel-operators.html for an overview.

In this appendix we explain the syntax of the navigation language of *e³value* , using the diagram in figure C.1 as running example.

C.1 Object names and identifiers

Each model element has a name that is given a value by the *e³value* editor but can be changed by you. In addition, each model element

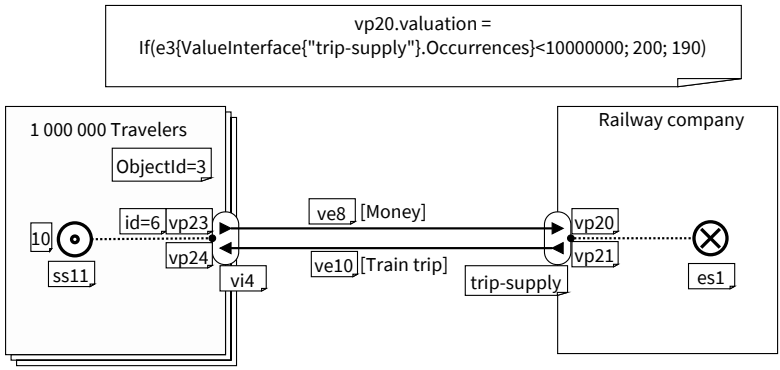

Figure C.1: A Railway company sells trips to Travelers.

has an object identifier, a unique number generated by the editor that cannot be changed by you.

```
{String Char} ::= {Printable}-["]-['#']-['{']-['}']
ObjectName ::= '"'{String Char}{String Char}*'"'
ObjectId  ::= {Digit}*
```

C.2 Properties

Each model element has properties, some of which are defined in the ontology and some of which may be defined by the user. Each property has a name.

```
PropertyName ::= '"'{String Char}{String Char}*'"'
```

A property can be referred to by its name, possibly prefixed with a uniquely idenfiying navigation expression that uses the unique object identifier or some navigation path.

```
<PropertyRef> ::=
   e3 '{'PropertyName'}' |
```

e3 '{' '#' ObjectId '.' PropertyName '}' |
e3 '{' <RelativeNavigationPath> '.' PropertyName '}' |
e3 '{' <AbsoluteNavigationPath> '.' PropertyName '}' |
e3 '{' <AbsoluteNavigationPath> '.'<RelativeNavigationPath> '.' PropertyName '}'

Two simple references to a property of a market segment are

e3{Size}
e3{#3.Size}

e3{Size} could be used in an expression to compute another property of the same market segment. e3{#3.Size} uniquely refers to the size of model element 3.

More complicated property references require a navigation path, explained below.

C.3 Absolute navigation paths

<AbsoluteNavigationPath>::=<AbsoluteNavigationStep>

<AbsoluteNavigationStep> ::=
<AbsoluteNavigationStepObjectName>'('')' |
<AbsoluteNavigationStepObjectNameU> '('ObjectName')'

<AbsoluteNavigationStepObjectName>::=Model

<AbsoluteNavigationStepObjectNameU>::=
Actor |
MarketSegment |
ValueObject |
Partnership

This allows the construction of property references such as the following (see figure C.1).

e3{Model() . Marketsegment(''Travelers'') . Size}
e3{Model() . Actor(''Railway company'') . Investment

C.4 Relative navigation paths

The following expressions navigate across relations in the ontology.

```
<RelativeNavigationPath> ::=
   <RelativeNavigationStep> |
   <RelativeNavigationPath>'.'<RelativeNavigationStep>

<RelativeNavigationStep> ::=
   <RelativeNavigationStepObjectName> '('')' |
   <RelativeNavigationStepObjectNameU> '(' ObjectName ')'
```

The relative navigatrion steps follow the links in the ontology. If a navigation step travels in the many-direction of a relationship, an object name is needed to disambiguate the step. A few examples will make this clear.

For example, from port vp20 in in figure C.1 we can construct the following relative navigation paths.

```
e3{upstream-transfer(''ve8'') . upstream-port() . Occurrences}
```

refers to the number of occurrences of the port at the other end of the transfer.

```
e3{upstream-transfer(''ve8'') . upstream-port() . ValueOffering() .
ValueInterface() . MarketSegment() . Name}
```

refers to the name of the market segment at the other end of the transfer.

Using absolute navigation, these same model elements could be referred to by the following expressions regardless where the expressions are used.

```
e3{#6 . Occurrences}
e3{model() . MarketSegment(''Travelers'') . ValueInterface(''vi4'') .
out-offering() . ValuePort(''ve23'') . Occurrences}
```

both refer to the number of occurrences of port ve23.

```
e3{Model() . MarketSegment(''Travelers'') . Name}
```

refers to the name of the Travelers market segment.

Bibliography

[1] M. van Alstyne. "The state of network organizations: A survey in three frameworks". In: *Journal of Organizational Computing and Electronic Commerce* 7.2&3 (1997), pp. 83–151.

[2] Soon-Yong Choi, Dale O. Stahl, and Andrew B. Whinston. *The Economics of Doing Business in the Electronic Marketplace*. Indianapolis, IN: MACMillan Technical Publishing, 1997.

[3] Colin Drury. *Management and Cost Accounting, third edition*. London, UK: Chapman and Hall, 1998.

[4] J. Gordijn and H. Akkermans. *Value webs. Understanding e-business innovation*. The Value Engineers, 2018.

[5] P. Kotler. *Marketing Management: Analysis, Planning, Implementation and Control*. Englewood Cliffs, NJ: Prentice Hall, 1988.

[6] J. Ramsay. "The real meaning of value in trading relationships". In: *International Journal of Operations and Production Management* 25.6 (2005), pp. 549–565.

[7] The Value Engineers. *A Catalog of Business Model Innovation Tactics*. `https://www.thevalueengineers.nl/download/a-catalog-of-business-model-innovation-tactics/`. 2020.

Index

E³ VALUE USER GUIDE

Any business is part of a network of suppliers, partners, infrastructure, platforms and competitors who collaborate and compete to provide value to customers. This is true not only for Spotify and Netflix but also for banking, farming and the manufacturing industry. If you fail to understand your ecosystem, you may discover one day that you are playing the wrong game. And if you introduce new technology, you need to understand its impact on your business ecosystem, on the value that your business adds, and on the revenue that you can generate from this.

E³ value is a language to build and analyze business models of your ecosystem. It has been successfully used to optimize business models for new technology in the music industry electricity supply, banking, news, telecommunications, and Internet service provisioning.

Jaap Gordijn is associate professor eBusiness at the Vrije Universiteit Amsterdam and managing partner at *The Value Engineers*. He has a comprehensive experience in designing and analyzing eco-systems for digital services.

Roel Wieringa is emeritus professor of Information Systems at the University of Twente, The Netherlands. He has written extensively on requirements engineering and conceptual modeling of software systems.

9 789082 852431

www.thevalueengineers.nl