Designing Controls for Network Organizations: A Value-Based Approach

ISBN 978 90 5170 861 5

Cover design: Crasborn Graphic Designers bno, Valkenburg a.d. Geul

This book is no. 422 of the Tinbergen Institute Research Series, established through cooperation between Thela Thesis and the Tinbergen Institute. A list of books which already appeared in the series can be found in the back.

VRIJE UNIVERSITEIT

Designing Controls for Network Organizations: A Value-Based Approach

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad Doctor aan
de Vrije Universiteit Amsterdam,
op gezag van de rector magnificus
prof.dr. L.M. Bouter,
in het openbaar te verdedigen
ten overstaan van de promotiecommissie
van de faculteit der Economische Wetenschappen en Bedrijfskunde
op woensdag 28 mei 2008 om 13.45 uur
in de aula van de universiteit,
De Boelelaan 1105

door

Vera Kartseva

geboren te Kharkiv, Oekraïne

promotor:	prof.dr.Y.H. Tan
	prof.dr.ir. R. Paans RE
copromotor:	dr. J. Gordijn

SIKS Dissertation Series No. 2008-11

The research reported in this thesis has been carried out under the auspices of SIKS, the Dutch Graduate School for Information and Knowledge Systems.

Preface

What does it take to accomplish a PhD research? One can name many things, but most importantly it takes involvement of many people around you. Many people have supported me during these four years and I would like to take this opportunity to express my gratitude to them.

Much credit goes to Jaap Gordijn, my PhD supervisor, who introduced me to this PhD project. I would like to thank him and my promotors Yao-Hua Tan and Ronald Paans for giving me this opportunity and for their support and guidance during these four years. They succeeded to inspire me for the subject as well as to give me enough freedom and intellectual support while making sure that the project progressed.

I would like to acknowledge the financial support of the FEWEB faculty and Information Systems group and, in particular, of the IT Audit Postdoctoral Education group at the VU and the EDP Audit group of KPMG. Support of these entities has made the research reported here possible and allowed it to be presented to the international community.

This research has been shaped by my involvement in several research projects, which provided me with additional insights and brought in more variety. I would like to thank all the participants of FRUX and ITAIDE project and, in particular, Rose-Marie Droës and Franka Meiland for their cooperation in the FRUX project.

In addition, I would like to thank Joris Hulstijn, Boriana Rukanova, Ziv Baida, Jianwei Liu, Zsofi Derzsi, Henry Dekker and the participants at the GREETING meeting for their interesting discussions and sharp remarks. I would also like to thank Zuzana Sasovova, Maura Soekijad, Frans Feldberg, Jeroen Walschots, Frank Derksen, Selmar Meents and other members of the Information Systems group for being great colleagues. Much credit also goes to the Bled conference community and to the E-dispuut members for their interest and feedback.

This book would not have been written if I had not started studying for a Masters degree at the Vrije Universiteit in the first place, which also required me to leave my home country and to come to the Netherlands. It was not easy, but challenging and inspiring. Therefore, next I would like to thank people who gave me this wonderful opportunity: Dieter Fensel and Hans Akkermans for seeing in me a potential Master student and Borys Omelayenko for his mentorship.

My friends played an important role during this time. I would like to thank Zhanna Serduykova, Maksym Korotkiy, Aiste Jonkute, Natasha Mostoviak, Sofia Lukjanchuk, Dima Tikhonov and other friends for having many fun moments together. I would also like to thank Corrie and Peter Schildwacht for their insights and advice.

The support from home has been very important for me during this time. As such, I would like to express my gratitude to my parents for being there for me at any moment, in spite of the distance, and last, but not the least, thanks go to my boyfriend Joost for his incredible patience, understanding and love.

Contents

1 Introduction **19**

1.1 Controls in network organizations . . . 19

1.1.1 Network organizations . . . 19

1.1.2 Opportunistic behaviour . . . 20

1.1.3 Network perspective on controls . . . 21

1.2 Research objective . . . 21

1.3 Design Requirements . . . 22

1.3.1 Control theory . . . 22

1.3.2 Requirements engineering . . . 24

1.4 The e^3*control* approach . . . 26

1.5 Research perspective . . . 29

1.6 Research outputs . . . 30

1.7 Research cycle . . . 30

1.8 Research activities . . . 32

1.8.1 Problem Analysis . . . 33

1.8.2 Solution Design . . . 33

1.8.3 Solution Implementation . . . 34

1.8.4 Implementation Evaluation . . . 36

1.8.5 Solution validation . . . 36

1.9 Research design criteria . . . 37

1.10 Research contribution . . . 40

1.11 Publications . . . 41

1.12 Outline of the research . . . 42

2 Designing controls in network organizations **43**
2.1 Inter-organizational controls 43
2.1.1 Internal control theory 44
2.1.2 Agency theory 48
2.1.3 Inter-organizational control theory 50
2.1.4 Management control theory 54
2.2 Design of controls 56
2.2.1 Value models 57
2.2.2 Value viewpoint on controls 58
2.3 Control patterns 60
2.4 Further steps 62
2.5 Summary 63

3 The e^3*control* methodology **65**
3.1 The e^3*value* methodology 65
3.1.1 Basic elements 65
3.1.2 Dependency path 67
3.1.3 Valuation 69
3.1.4 The e^3*value* ontology 71
3.1.5 Comparison with other approaches 71
3.2 The e^3*control* design framework 73
3.2.1 Subjective view on sub-ideal behaviour 74
3.2.2 A need to extend e^3*value* 75
3.3 Constructs of the e^3*control* ontology 76
3.3.1 Sub-ideal value objects 78
3.3.2 Sub-ideal value transfers 78
3.3.3 Empty and incorrect sub-ideal value objects 78
3.3.4 Liability tokens 78
3.3.5 Sub-ideal value port 79
3.3.6 Sub-ideal value interface 79
3.3.7 Sub-ideal dependency segments 80
3.3.8 Sub-ideal dependency paths 80
3.3.9 Sub-ideal actors 80

3.4 Modelling rights and evidence 80
3.5 An extended value object ontology 81
3.5.1 Resources 82
3.5.2 Rights 82
3.5.3 Transfer and Conversion activities 83
3.5.4 Evidence documents 83
3.6 Rights and evidence in e^3*value* 84
3.6.1 Rights and custody in one value transfer 84
3.6.2 Splitting transfer of rights and custody 85
3.6.3 Criteria for distinguishing a value transfer 85
3.7 Summary 87

4 Controls in the private sector **89**
4.1 Research Context 92
4.2 Step 1. Ideal value model 92
4.3 Step 2. Sub-ideal value model 92
4.4 Step 3. Modelling controls 96
4.4.1 Solution 1: Collecting evidence from listeners 96
4.4.2 Solution 1: Information systems viewpoint 98
4.4.3 Solution 2: Sharing information with advertisers 101
4.5 Discussion 103
4.6 Lessons learned 105
4.7 Summary 107

5 Controls in the public sector **109**
5.1 Research Context 110
5.2 Extraordinary health care system 110
5.2.1 Step 1. Ideal value model 111
5.2.2 Step 2. Sub-ideal value model 113
5.2.3 Step 3. Control of indication centres and care providers 115
5.3 Personal budget and Social Chart 118
5.3.1 Step 1. Ideal value model 118
5.3.2 Step 2. Sub-ideal value model 120
5.3.3 Step 3. Controls: Social Chart 120

5.4 Discussion . . . 123
5.5 Lessons learned . . . 125
5.6 Summary . . . 126

6 Elicitation of control patterns 129
6.1 Patterns elicitation method . . . 130
6.2 Domain of inter-organizational controls . . . 132
6.3 Commonality - variability analysis . . . 133
6.3.1 Grouping criteria . . . 134
6.3.2 Dealing with delegation . . . 135
6.3.3 Deriving the patterns from control sub-domains . . . 135
6.4 Control patterns . . . 138
6.5 Classification of control patterns . . . 138
6.6 Summary . . . 139

7 Patterns vocabulary 141
7.1 A control pattern template . . . 142
7.2 Ideal situation vocabulary . . . 143
7.2.1 Value view . . . 143
7.2.2 Process view . . . 144
7.3 Sub-ideal situation vocabulary . . . 145
7.3.1 Value view . . . 145
7.3.2 Process view . . . 145
7.4 Reviewing control terminology and principles . . . 147
7.4.1 Models of internal controls . . . 148
7.4.2 Models of inter-organizational controls . . . 149
7.4.3 Limitations of existing work . . . 150
7.4.4 The control activities and control principles vocabulary . . . 151
7.5 Control activities vocabulary . . . 151
7.5.1 Verify activity . . . 152
7.5.2 Witness activity . . . 153
7.5.3 Testify activity . . . 154
7.5.4 Promise activity . . . 154
7.6 Control principles vocabulary . . . 155

7.6.1 Witnessing Principles . . . 156
7.6.2 Verification Principles . . . 158
7.6.3 Promising Principles . . . 160
7.6.4 Testification Principles . . . 161
7.6.5 Pre-execution Principles . . . 161
7.7 Summary . . . 163

8 Procedural patterns **165**
8.1 Partner Screening . . . 165
8.1.1 Context . . . 166
8.1.2 Problem . . . 166
8.1.3 Solution . . . 166
8.1.4 Example . . . 168
8.1.5 Variations . . . 170
8.1.6 Related patterns . . . 170
8.2 Proper Contracting pattern . . . 170
8.2.1 Context . . . 171
8.2.2 Problem . . . 171
8.2.3 Solution . . . 171
8.2.4 Example . . . 173
8.2.5 Variations . . . 174
8.3 Execution Monitoring pattern . . . 175
8.3.1 Context . . . 175
8.3.2 Problem . . . 175
8.3.3 Solution . . . 175
8.3.4 Example . . . 177
8.3.5 Variations . . . 177
8.3.6 Related patterns . . . 179
8.4 Execution Confirmation pattern . . . 179
8.4.1 Context . . . 179
8.4.2 Problem . . . 180
8.4.3 Solution . . . 180
8.4.4 Example . . . 182
8.4.5 Variations . . . 182
8.5 Summary . . . 185

9 Contractual patterns **187**

9.1 Penalty pattern . . . 187

9.1.1 Context . . . 187

9.1.2 Problem . . . 189

9.1.3 Solution . . . 189

9.1.4 Example . . . 191

9.1.5 Related patterns . . . 193

9.2 Incentive pattern . . . 193

9.2.1 Context . . . 193

9.2.2 Problem . . . 193

9.2.3 Solution . . . 193

9.2.4 Example . . . 196

9.2.5 Related patterns . . . 197

9.3 Summary . . . 197

10 Delegation patterns **199**

10.1 Delegation Box . . . 199

10.2 Simple Delegation pattern . . . 201

10.2.1 Context . . . 201

10.2.2 Problem . . . 202

10.2.3 Solution . . . 202

10.2.4 Example: Certification . . . 202

10.3 Testifying Chain pattern . . . 206

10.3.1 Context . . . 206

10.3.2 Problem . . . 206

10.3.3 Solution A: Low trust . . . 206

10.3.4 Solution B: High trust . . . 208

10.3.5 Example . . . 208

10.4 Summary . . . 208

11 The e^3*control* design framework for patterns **211**

11.1 Step 1: Design an ideal situation . . . 211

11.1.1 Guidelines . . . 213

11.1.2 Example . . . 213

11.1.3 References . . . 213

11.2 Step 2: Design a sub-ideal situation . . . 213

11.2.1 Guidelines . . . 214

11.2.2 Example . . . 215

11.2.3 References . . . 215

11.3 Step 3: Select a control pattern . . . 216

11.3.1 Step 3.1. Identify ideal and sub-ideal primary and counter actors, value objects, and activities . . . 216

11.3.2 Step 3.2. Match the sub-ideal activity to a pattern . . . 218

11.3.3 Guidelines . . . 219

11.3.4 Example . . . 220

11.3.5 References . . . 221

11.4 Step 4: Apply the control pattern . . . 221

11.4.1 Step 4.1. Match context and problem . . . 221

11.4.2 Step 4.2. Design a value model that solves a control problem . . . 223

11.4.3 Step 4.3. Design a process model that solves a control problem . . . 223

11.4.4 Guidelines . . . 223

11.4.5 Example . . . 225

11.4.6 References . . . 227

11.5 Summary . . . 228

12 Renewable electricity in UK **229**

12.1 Research Context . . . 230

12.2 The first e^3*control* cycle: Non-tradable ROCs . . . 230

12.2.1 Step 1: Ideal situation . . . 230

12.2.2 Step 2: Sub-ideal situation . . . 233

12.2.3 Step 3: Select a control pattern . . . 235

12.2.4 Step 4: Apply the Penalty and Incentive patterns . . . 237

12.3 The second e^3*control* cycle: Tradable ROCs . . . 241

12.3.1 Step 1: Ideal value and process models . . . 242

12.3.2 Step 2: Sub-ideal value and process models . . . 244
12.3.3 Step 3: Select a control pattern . . . 246
12.3.4 Step 4: Apply the Execution Monitoring pattern . . . 246
12.4 The size of the buy-out fee and the buy-out premium . . . 248
12.4.1 Acquiring ROCs together with green electricity . . . 248
12.4.2 Buying ROCs separately . . . 249
12.4.3 ROC Fee as a penalty . . . 249
12.5 Discussion . . . 250
12.6 Lessons Learned . . . 250
12.7 Summary . . . 252

13 Beer export **253**
13.1 Research Context . . . 253
13.2 Excise collection within the EU . . . 254
13.3 Case study structure . . . 255
13.4 Tampered AAD scenario . . . 256
13.4.1 Step 1: Design ideal situation . . . 256
13.4.2 Step 2: Design sub-ideal situation . . . 260
13.4.3 Step 3: Select a control pattern . . . 260
13.4.4 Step 4: Apply the Penalty pattern . . . 265
13.5 EMCS scenario . . . 270
13.6 Virtual Beer scenario . . . 273
13.6.1 Step 1: Design ideal situation . . . 274
13.6.2 Step 2: Design sub-ideal situation . . . 274
13.6.3 Step 3: Select a control pattern . . . 276
13.6.4 Step 4: Applying patterns . . . 280
13.7 Future scenario with TREC . . . 281
13.7.1 Innovative Trade Procedure . . . 284
13.7.2 Analyzing the TREC scenario with patterns . . . 285
13.8 Discussion . . . 285
13.9 Lessons Learned . . . 289
13.10 Summary . . . 291

14 Conclusions and future research **293**
14.1 Key research outputs 293
14.1.1 The network perspective 294
14.1.2 The economic value viewpoint on controls 295
14.1.3 The e^3*control* ontology 299
14.1.4 The e^3*control* patterns 300
14.2 Reflection on experience with the e^3*control* approach 301
14.3 Managerial implications 303
14.4 Research limitations and future research 304
14.5 Future of the design of controls in networks 306

Appendices **309**

A Controls principles **311**

B UML Activity Diagrams **315**

Chapter 1

Introduction

In this chapter we introduce the topic of our thesis: the design of controls in network organizations. We discuss the concept of a network organization, as dealt with in this work. Furthermore, we introduce the notions of opportunistic behaviour, control mechanism, and the network perspective on controls. We also give an overview of the e^3*control* methodology, which is developed in this thesis. In addition, a short review is made of the principles of conceptual modelling and the economic value perspective which underpin this methodology. Finally, we discuss the research questions, research methodology, research outcomes and contributions made by this thesis.

1.1 Controls in network organizations

1.1.1 Network organizations

Organizations have increasingly formed themselves into networks. In mobile communications sector companies worked together to develop standards such as GSM and WAP [Bekkers et al., 2002]. Toyota has led the field in mastering tight inter-firm collaboration with its suppliers, and has hereby managed to deliver high quality products at reasonable prices for many years [Schonberger, 1986]. These examples are no exceptions, and they have become common in organizing business.

Companies pursue network strategies for various reasons. These reasons include access to new markets, increased efficiency, lower risk of R&D, access to specialized and complementary competencies, the ability to serve individual consumer needs, hedge against lagging behind technology, setting standards, and obtaining subsidies or grants [Man, 2004].

There are many definitions of network organizations in business literature[1]. What we define a network organization as a set of independent, privately or publicly owned companies with their own objectives (e.g. profit making or strategic objectives). These companies are bound to some

[1] The phenomena of network organization is referred to with many different terms, including ‘network organization’, ‘inter-organizational relation’, ‘joint venture’, ‘small firm network’, ‘value-adding partnership’,‘virtual corporation’, ‘value network’, ‘value web’ and ‘value constellation’ (see [Alstyne, 1997])

short- or limited-term contractual agreements aimed at a targeted joint business activity, such as the joint delivery of some service to a final customer.

Decision making in networks is not so much based on one authority, as in hierarchies, but is more distributed, as in markets [Bakos, 1998]. Decisions in networks are not imposed by one central organization, but are negotiated between network participants. Such negotiations are necessary since the organizations participate in networks voluntarily. By participating in networks, companies want to achieve their business objectives. So, if a company is not satisfied with their gain from the network, it can leave the network. As a result, the network can disappear. One of the reasons why companies exit the network is the *opportunistic behaviour* of other network participants.

1.1.2 Opportunistic behaviour

Parties in networks are vulnerable to the opportunistic behaviour of others, mainly due to the close link both on business and communication levels. Networks are often characterized by high-level integration and inter-connection between individual firms. Such integration and inter-connection takes place both on the communication level, by creating tight technology-enabled information links between firms, as well as on the business level, by entering into partnerships in order to profit from the sharing of resources and knowledge.

On the communication level, the seamless information links with network partners may make the company more vulnerable to their opportunistic behaviour. For example, network partners may get easier access to confidential company information. This threat is relevant for both business-to-business and business-to-consumer transactions. Especially the electronic business-to-consumer transactions are vulnerable to the communication-level risks, e.g. credit card fraud.

On the business level, an company's ability to meet customer's expectations increasingly depends on the performance of the network partners. For example, in order to deliver an information service via a mobile network, the service provider has to cooperate with a mobile operator. However, if the mobile operator experiences technical problems and is not able to deliver the service properly, it will have a negative effect on the service provider's revenue. So a company's revenue is vulnerable to the opportunistic behaviour of the network partners.

Yet another factor to be taken into account is the digitalization of the network economy. Many of today's products are *knowledge-based* such as digital music or Internet newspapers. These products obey a law of increasing returns: once the costs of making the first digital copy have been invested, the reproduction costs are almost zero [Tapscott et al., 2000, p. 6]. Although this results in huge potential profits for the producer of the product, their *collection* from the distribution network is an enormous problem. The reason is that the producers cannot control how many products are sold by their network partners, because these partners can also easily copy the product and sell it without the producer knowing about it. So music or software industries suffer enormous losses due to illegal downloads and pirate copying.

In general, the losses that stem from opportunistic actions are enormous. Losses from fraud alone account for around 5% of the US GDP, a staggering $652 billion [ACFE, 2006]! In Europe, losses from a VAT-tax fraud account for 100 billion euros loss of revenue per year

[Houlder, 2007]. Legal music downloads are gaining in popularity, yet they were outnumbered by their illegal counterparts ten-to-one in 2006, as according to a report by an NPD Group survey of consumers [NPD, 2007].

1.1.3 Network perspective on controls

To be sustainable, a network organization needs mechanisms to govern and control the interaction among network participants and to mitigate opportunistic behaviour. In most cases, this interaction is governed by contractual arrangements and implemented through procedures and regulations. *Control mechanisms* or just *controls* constitute a part of the contractual agreements. By the term 'inter-organizational controls' we understand *measures one or more parties can use to prevent, detect or correct opportunistic behaviour of its counter parties*. The design of such controls is the subject of this thesis.

It is essential to understand that we are explicitly examining the opportunistic behaviour of the *external parties* of an organization, and not of its employees. In addition, the controls we are considering take into account the activities of the external parties. We call this a **network perspective on controls**.

The network perspective is important, because if networks are to be sustainable controls must ensure that no network participant behaves opportunistically. As a result, we assume that one or several network participants will introduce controls to protect themselves from the (potential) opportunistic behaviour of other participants. In addition, controls in the networks are designed during a collaborative decision making process, as describe above.

1.2 Research objective

The objective of this research is to develop a methodology of designing controls for network organizations. We define the objective as follows:

- *To develop a methodology that would support human analysts in designing controls against opportunistic behaviour in networks of organizations.*

Design methodologies for network organizations have received considerable attention in recent years, e.g. [Osterwalder and Pigneur, 2002], [Tapscott et al., 2000], [McCarthy, 1982], [Gordijn and Akkermans, 2003], [Pateli and Giaglis, 2004]. However, most of the methodologies focus on the design of networks from the efficiency and profitability perspective rather than addressing the issue of control.

Controls in inter-organizational relations attract a lot of interest, especially in management accounting research into strategic alliances and joint ventures (e.g. [Das and Teng, 1998], [Dekker, 2004]). However, this research is mostly theory-oriented and not much has been done on designing inter-organizational controls.

In general, many alternative views on the nature of controls exist. Inter-organizational controls have been studied in many fields, including public administration, law, business sciences and information systems. Each field addresses specific aspects of controls, but none of the field alone is sufficient to address the issue completely. For example, proper legislation which prohibits illegal downloads is required to tackle the problem of illegal music distribution via Internet. However, proper legislation is not enough, if there is no technology to support it.

In this research, we address controls from the perspective of two disciplines: *internal control* and *requirements engineering*. The internal control field provides a starting point to define the concept of control and to get insights into how the controls are designed in hierarchical organizations. Taken in the perspective of other theories, such as the *agency theory*, the internal control field also provides some ideas about how controls in networks can be designed. The requirements engineering field provides a more solid theoretical framework on how to develop the *design* methodology.

1.3 Design Requirements

In this section we briefly discuss the two pillars of this research: *internal control* and *requirements engineering*. We explain how these two disciplines can be used to understand the design of controls in network organizations and, from that, we derive the requirements for our design methodology. This allows us to formulate our research questions.

1.3.1 Control theory

Internal control is a well-established field of practice, which mainly concerns with procedural measures to safeguard assets of an organization. There are many different definitions of internal control. [Simons, 2000] defines it as the *set of procedures* that dictate how and by whom information should be recorded and verified in order to provide the checks and balances to ensure that assets are safeguarded and the information collected and processed by the accounting system is accurate. Some use a somewhat broader definition, for example [Knechel, 2001] and [Hayes et al., 2005] see internal control as a process that is designed by management to reach reasonable assurance that the following objectives will be met: 1. improvement of decision making's effectiveness and operational efficiency; 2. increase reliability of information; 3. foster compliance with laws, regulations and contractual obligations; 4. safeguarding of assets. In general, internal control is described in Dutch literature within Internal Control/ Administrative Organization (IC/AO) discipline [Starreveld et al., 1994] and in Anglo-Saxon literature within the Accounting Information Systems (AIS) discipline [Romney and Steinbart, 2006], [Hollander et al., 1999].

By our definition of *internal control theory*, maintained henceforth, we refer to internal control as it is described in [Romney and Steinbart, 2006], [Hollander et al., 1999] and [Starreveld et al., 1994], which also corresponds to the definitions given above. Internal controls described in this literature mainly refer to control measures of *procedural nature* implemented *inside* an organization. Some everyday examples of typical internal control problems are the theft of cash or inventory, forging accounting records or making errors in administration. Such occurrences results in inaccurately reported profits and can be aimed at hiding of embezzlement of the organization's assets. The usual internal controls to mitigate such control problems include segregation of duties, establishing audit trail, proper authorization, independent verification and access restriction (see Chapter 2).

The recent trend in practice is to focus on a general concept of *management control*, of which internal control is only a part. Management control is broadly defined as all activities and decisions taken by management to ensure that employees behave in such a way that they increase the probability that the organization will achieve its goals [Merchant and van der Stede, 2007]. This may include procedural measures, such as internal controls, but also value-based propositions, that can be taken from agency-theory and transaction cost economics. In this work we in particular employ *agency theory* to get understanding of inter-organizational and value-based aspects of controls.

Relation to the network perspective

The internal control theory focuses on *internal* controls rather than on those in a network context. The inter-organizational control problems - the ones caused by the opportunistic behaviour of parties other than the company's employees - are not explicitly addressed in the internal control theory. They are assumed to be mitigated by the internal controls, which are not specifically designed for that purpose.

The network perspective on controls contrasts with the traditional practice of the internal control theory, which implies that an organization manages only the activities within the boundaries of the organization. In addition, the internal control theory makes no explicit distinction with regard to who causes the risk: an internal employee or an external party. In principle, from the network perspective we only consider controls against the opportunistic behaviour of *counter parties*. As a result, such controls must take into account the activities of other parties in the network. For example, regulation between a company's internal activities and the external activities of its counter parties becomes important. So, the inter-organizational controls have an effect outside the boundaries of any one company.

Many of the inter-organizational controls that we describe in this thesis are known to the internal control field though not explicitly outlined as inter-organizational controls. One of our contributions in this thesis is to remedy that situation.

It is not easy to apply the internal control theory to designing controls from the network perspective, so, some other theories are also considered. In particular, we use *agency theory* and *management control theory* as sources to provide a general conceptual framework and classifications of inter-organizational controls. For example, as with the agency theory, we distinguish between two parties: an *agent*, a party that behaves opportunistically, and a *principal*, a party that suffers from this opportunistic behaviour and wishes to control it. Similar to management control theory, we distinguish between *procedural* against *contractual* controls and *ex-ante* against *ex-post* controls. This will be explained further in the next section.

1.3.2 Requirements engineering

Networks of organizations are characterized by a high level of Information Systems (IS) usage. IS have many advantages, such as increasing the speed of communication or reducing transaction costs [Tapscott et al., 2000]. Nowadays, as more and more networked information systems come in place, controls become an intrinsic part of information systems development [ISACF, 2005]. We therefore expect that the final target in the design of controls in a network organization will often be the adaptation of an information system. Such an information system would enable these controls. This is similar to the internal controls, which are incorporated into the organization's accounting information system [Hollander et al., 1999].

In this thesis, *the design of controls is approached as part of a requirements engineering task*. Requirements engineering is the process of developing requirements for IS through an iterative co-operative process of analyzing the problem, documenting the resulting observations, and checking the accuracy of the knowledge gained [Loucopoulos and Karakostas, 1995]. Requirements engineering consists of eliciting, representing and evaluating the software technical requirements for IS artifacts at hand. Requirements for information systems can be very complex [Sommerville and Sawyer, 1997], because, for instance, of the multiple contradictory requirements involved. Therefore, requirements engineering research develops techniques that help to tackle these issues of complexity.

Two important components are common to requirements engineering techniques. Firstly, the concept of *multiple viewpoints* and, secondly, *(graphical) conceptual modelling*. We will now go on to discuss how these two components are used in this work.

Value viewpoint

The design of inter-organizational controls can be very complex, because, amongst other things, different perspectives of the multiple stakeholders are involved. Controls in networks are not usually imposed on the network by one central organization as is the case in a single enterprise, but are negotiated by all network participants, e.g. business analysts, system developers, CIOs, CEOs. As in a free market system, the decision making process within networks is spread out and does not depend on one authority, as in a hierarchy. Stakeholders often have different views on control problems and the control mechanisms that solve them. For example, a financial manager will focus more on the economic implications of a solution, rather than an IS engineer who is more interested in its technical details. Such multiple perspectives add extra complexity to the decision-making process.

Multiple viewpoints have been put forward in requirements engineering as mechanisms in dealing with this multi-perspective problem. By describing one system from several viewpoints, complicated requirement issues can be broken down into self-contained perspectives, which can be addressed and decided independently of each other [Finkelstein et al., 1992].

Amongst other things, requirements analysis distinguishes between a *process viewpoint*, which describes business processes, and an *information system viewpoint*, which describes information systems that enable and support the business processes. Recently, many researchers emphasize the necessity of taking *business requirements* into account for information systems. To address this issue, [Gordijn and Akkermans, 2003] have introduced the *economic value viewpoint*, which focuses on the way economic value is created, exchanged and consumed in a multi-actor network. In this work, we emphasize the development of the *economic value viewpoint* for controls.

The internal control theory, the development of which dates back to the early days of information systems engineering, only considers the design of controls from a process and information system viewpoint. We consider it important to add the value viewpoint to the design of inter-organizational controls for several reasons. They are discussed in detail in Chapter 2. In short, the following arguments are put forward.

Firstly, in a network, the provision of *controls can be seen as a commercial service with added value* and have their price tag. This has been observed in a number of case studies [Kartseva et al., 2004a, Kartseva et al., 2005a, Kartseva et al., 2005b, Kartseva et al., 2006b, Kartseva et al., 2006c]. The second argument is that *many control instruments (e.g. evidence documents, incentives) have inherent value aspects*. Documents, used for control reasons, often have an economic value and can be traded. An example of such a document is the Bill of Lading, which is a document in international trade procedures that must be presented by the buyer to the carrier to claim goods [Kartseva et al., 2004a]. From the control perspective, the Bill of Lading is proof that the buyer has paid for the goods. On the other hand, typically, Bills of Lading for overseas trade are usually traded many times before the shipped goods are actually claimed. The ownership of goods is also transferred every time a Bill of Lading is traded, which makes it an object of value. The last argument is that *the value viewpoint provides a good rationale for procedural controls*. By introducing a value viewpoint, we provide a high-level model of controls from the economic perspective, without focusing on procedural details. Controls are primarily about safeguarding objects of value and the value perspective points out which objects of value should be guarded.

The development of the value viewpoint on controls is a major contribution given by this thesis and, in addition, we also consider some aspects of integration with the process perspective on controls. This is mainly done within the control patterns described in Chapters 8 and 9. The information systems perspective on controls is out of the scope of this work.

Conceptual modelling

Another component taken from requirements engineering is *conceptual modelling*. Conceptual models consist of a number of symbol structures and symbol structure manipulators which are supposed to correspond to the conceptualization of the world by human observer [Borgida et al., 1982]. This view appears to underly the work on requirements engineering methodologies. Examples of well-known conceptual modelling techniques are Data Flow charts and Unified Modelling Language (UML) diagrams [Fowler and Scott, 2000], [Hollander et al., 1999]. Over the years, the requirements engineering field has developed rigorous and novel model-based methods for conceptualization. Conceptual modelling and graphical diagramming methods have been elaborated into a fine art at a level of sophistication not found in other disciplines [Akkermans and Gordijn, 2006] in business science.

In particular, conceptual modelling is useful for designing controls in a network environment. The design of such controls is more complex than in a hierarchical organization with a central authority, since the design process requires negotiations between multiple stakeholders. The stakeholders often have different views on control problems and solutions and different interests, which, when communicated in normal language, may lead to incomplete and ambiguous statements [Malone, 1987]. The advantage of conceptualization is that the conflicts, which can occur during the negotiation process are made explicit, and this helps stakeholders to resolve them at an early stage of development. For further discussion on the conceptual modelling see Chapter 2.

Internal control practitioners utilize thorough conceptualization of business processes and information systems to get an understanding of the control problems and the necessary controls (see e.g. [Romney and Steinbart, 2006]). What is innovative in our work is that we introduce conceptualization of controls from the *value* perspective. In addition, as will be explained later, we utilize the concept of *patterns* for conceptual modelling.

1.4 The e^3*control* approach

In the previous section we have identified several requirements which we believe a design methodology for inter-organizational controls should comply with. In short, they include incorporating the economic value perspective, the network perspective and conceptual modelling in our design methodology. In order to address the research objective and to take into account these requirements, we have defined the following research questions:

- ***Research Question 1:*** *What steps are needed to design controls in network organizations from the economic value perspective?*
- ***Research Question 2:*** *What constructs are needed to describe the control problems and control mechanisms from the economic value perspective?*

To address the research questions, we introduce the e^3*control* ontology and the e^3*control* design framework. The **e^3*control* design framework** is a step-wise method which guides the design of control mechanisms and therefore addresses the first research question. It starts by analysing of the business network, then proceeds to analyse control problems in the network, and finishes with the design of control mechanisms. The framework is shown in Figure 1.1. See Chapter 3 for more details.

The **e^3*control* ontology** provides constructs to describe control problems and control mechanisms from an economic value perspective and therefore addresses the second research question. In general, an *ontology* is a common method of representing semantic conceptual models - explicit formal specifications of the terms in the domain and the relationships between them [Gruber, 1994]. Ontologies have become common on the World-Wide Web and range from large taxonomies categorizing Web sites (such as on Yahoo!) to categorizations of products for sale and their features (such as on Amazon.com). Wide-spread general-purpose ontologies are emerging as well, for example, the United Nations Development Program and Dun & Bradstreet

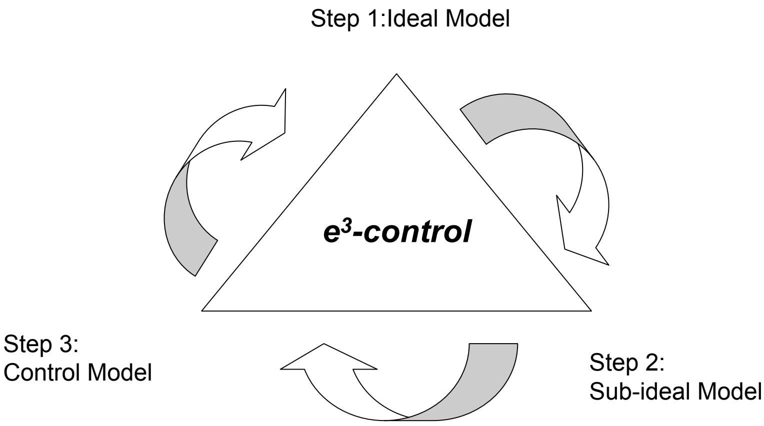


Figure 1.1: The *e*^{3}*control* design framework

have combined their efforts to develop the UNSPSC ontology which provides terminology to describe products and services (www.unspsc.org) [Noy and McGuinness, 2001]. Modern definitions of ontology (see e.g. [Borst et al., 1997]) emphasize that there must be agreement on the conceptualization that is specified: 'An ontology is a formal specification of a shared conceptualization'.

An ontology defines a common vocabulary for researchers who need to share information in a domain. It includes machine-interpretable definitions of basic concepts in the domain and the relationships between them. Some prime reasons for developing an ontology are: to share a common understanding of the structure of information among people or software agents; to enable reuse of domain knowledge; to make domain assumptions explicit; to separate domain knowledge from operational knowledge; and to analyze domain knowledge [Noy and McGuinness, 2001]. Some of these goals are also relevant for the design of inter-organizational controls.

The *e*^{3}*control* ontology is based on the existing *e*^{3}*value* ontology, introduced by [Gordijn and Akkermans, 2003]. It represents concepts, which describe a network organization as a set of *actors* that exchange *objects of economic value* with each other. By instantiating the *e*^{3}*value* ontology, one can create *e*^{3}*value* models (or just *value models*), which describe value transfers between network participants. The value models describe the network organization from an economic value perspective. Although there are several other suitable ontologies, which describe the economic value perspective, in this thesis we use the *e*^{3}*value* ontology, as motivated in section 3.1.5.

However, since the *e*^{3}*value* ontology does not explicitly consider control aspects, such as opportunistic behaviour we have to extend it to specifically include the concepts needed to model the effects of opportunistic behaviour on value transfers. The resulting ontology is known as the *e*^{3}*control* ontology and because it is an extension of the *e*^{3}*value* ontology, to create graphical *e*^{3}*control* models we can still use the software tool support developed for *e*^{3}*value* .

The *e*^{3}*control* ontology and the *e*^{3}*control* design framework can be used to design controls in networked organizations. However, the design process still requires a vast amount of knowledge on inter-organizational controls *themselves*. This leads us to the third research question, which is:

- ***Research Question 3:*** *How to represent the recurrent control problems in networks of organizations and what are the accepted and proven control mechanisms that mitigate these problems?*

To make this knowledge on available, we propose a *library of control patterns*. The patterns approach is well-known for structuring complex design knowledge. The patterns are defined as descriptions of common solutions for recurrent problems [Gamma et al., 1995]. The **e^3*control* patterns** represent problem-solution pairs of inter-organizational control problems and mechanisms, and describe control problems and mechanisms from both the value and the process viewpoints. The value viewpoint focuses on the economic value aspects of controls, while the process viewpoint focuses on their procedural aspects. The patterns can be used in the design process by the business analyst to model new controls. Typically these new controls are composed from instantiation of a pattern.

In general, the patterns capture *accepted* design knowledge [Gamma et al., 1995]. The knowledge captured by e^3*control* patterns is readily available rather than invented in this work. We focus on structuring the existing knowledge about control problems and control mechanisms in patterns.

The e^3*control* patterns also contribute to the second research question. In addition to the e^3*control* ontology, they contain a *vocabulary* which describes the additional concepts needed to design the controls.

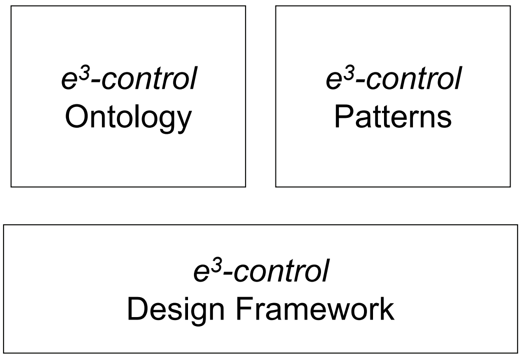


Figure 1.2: The components of the e^3*control* approach

To summarize, the e^3*control* ontology, the e^3*control* patterns and the e^3*control* design framework constitute the e^3*control* approach, see Figure 1.2. This approach is unique because it is based on an *economic value perspective*. It is the transfer of objects of economic value in a network that has to be controlled in the first place. In addition, it assumes a *network perspective*, which implies that parties in a network of enterprises introduce controls to reduce any losses caused by the opportunistic behaviour of other network members. This is in contrast to the traditional internal control theory, which focuses on internal controls from a process-oriented view. Finally, the e^3*control* approach provides a structured way of designing controls by employing such techniques as *ontologies* and *patterns*.

1.5 Research perspective

The type of our research is *design-oriented* and *inter-disciplinary*; as already explained, we focus on two fields: *internal control* and *requirements engineering*. The internal control discipline and the focus on the economic value relate this research to the field of *business science*. The requirements engineering perspective links this research to *computer science*.

In business science, our research can be positioned in the paradigm of *design science*. As stated by [Hevner et al., 2004], design science seeks to extend the boundaries of human and organizational capabilities by creating new and innovative *artifacts*. Design is a key activity in fields like architecture, engineering, medicine, and law [van Aken, 2004].

Design science is often contrasted with the *explanatory sciences*, like physics, sociology and economics. The mission of an explanatory science is to describe, explain and possibly predict observable phenomena within its field [Simon, 1976]. Explanatory sciences are engaged in a quest for solving pure knowledge problems, such as why one country is wealthier than another or why one information system is more user-friendly than another. On the other hand, design sciences develop prescription-driven models, that can be used in an instrumental way to design solutions for management problems [March and Smith, 1995]. Design sciences focus on the knowledge to be used in designing solutions for existing problems, such as the design of user-friendlier information systems or the design of a governance system to reduce tax fraud. 'The mission of a design science is to develop knowledge for the design and realization of an artifact, i.e. to solve construction problems, or to be used in the improvement of the performance of existing entities, i.e. to solve improvement problems' [van Aken, 2004].

In business science, the dominating research paradigm tends to be the explanatory science. Nevertheless, more and more researchers in business science address the value of the design science in the business domain [Hevner et al., 2004], [March and Smith, 1995], [van Aken, 2004]. One of the driving forces behind the development of design-oriented research in business sciences is the *utilization problem* [Susman and Evered, 1978]. There are serious doubts about the actual relevance of present-day management theory as developed within the exploratory research paradigm. In their study of American management education, [Porter and McKibbon, 1988] point out that the business world is ignoring the research coming from Business Schools. 'Management theory is either scientifically proven, but then too reductionistic and hence too broad or too trivial to be of much practical relevance, or relevant to practice, but then lacking sufficient rigorous justification [van Aken, 2004] p. 221. As a result, [van Aken, 2004] suggests that the results of design science, with its ultimate focus on prescription-driven solutions will contribute to the relevance of the knowledge developed by the academics in the business sciences. The importance of design science can be confirmed by the recent developments in the field of business modelling [Osterwalder and Pigneur, 2002], [Tapscott et al., 2000], [Weill and Vitale, 2001], [Gordijn and Akkermans, 2003], where the design research paradigm is dominating.

Design-oriented research is not new in the field of *computer science*, although this community does not specifically identify it as 'design science'. Requirements engineering journals and conferences contain many contributions that can be described as design science research [Wieringa et al., 2005], [Akkermans and Gordijn, 2006]. Moreover, proposals for different innovative conceptual modelling techniques and methods for requirements engineering are *core* contributions given by the requirements engineering research community. As [Wieringa et al., 2005] states, design in itself is "useful for other engineering researchers even if the design is not validated, because they could replicate the technique and validate its properties, or use it to solve their own problems, which might be problems the designer of the technique had not considered".

To summarize, there is no common view on what design-oriented research is. Design-oriented research in business science, known as *design science*, is quite a new phenomenon compared to the explanatory type of research. Design research in computer science is quite a developed field. Further to define our research process we incorporate the best findings of both communities.

1.6 Research outputs

The usual research product of design-oriented research is the heuristic prescription, *technological rule* or solution concept [van Aken, 2004]. These outputs are often called 'artifact' in design science literature and can be of four types: constructs, models, methods and implementations [March and Smith, 1995]. *Constructs* constitute a vocabulary of a domain. A *model* is a set of propositions or statements expressing the relationship between constructs. A *method* is a set of steps used to perform a task. An *instantiation* is the realization of an artifact in its environment.

In this research, we make use of four design outputs:

1. Constructs: The e^3*control* ontology in Chapter 3 represents a collection of constructs required to describe a domain. In addition, the *vocabulary* in Chapter 7, which is a part of the e^3*control* patterns library, describes additional constructs of a domain.

2. Models: The e^3*control* patterns in Chapters 8 and 9 represent models which describe the relationships between the constructs. These relationships are of a prescriptive nature, meaning that they express how the constructs should be used to design correct controls.

3. Method: The e^3*control* design framework in Chapter 11 is a method that sets up steps to perform the process of controls design.

4. Implementation: Implementation artifacts are models that are developed as a result of the instantiation of other e^3*control* artifacts in case studies. These models are described in the case studies in Chapters 4, 5, 12 and 13.

1.7 Research cycle

The process of design research can be conducted in many different ways. A generally accepted view in design science is that the process comprises of two main research activities: *build* and *evaluate* [March and Smith, 1995], [Hevner et al., 2004]. Building is a process of constructing an artifact for a specific purpose and evaluation is the process of determining how well the artifact performs.

Requirements engineering literature provides a different approach on how design-oriented research should be conducted [Wieringa et al., 2005], [Akkermans and Gordijn, 2006]. In particular, it identifies the *engineering cycle* as the logical structure or engineering activity [Wieringa et al., 2005]. In our view, this approach is more solid, since it is based on decades of practising design-oriented research in engineering and its major assumptions have been tested empirically, e.g. by [Cross, 2001] and [Witte, 1972].

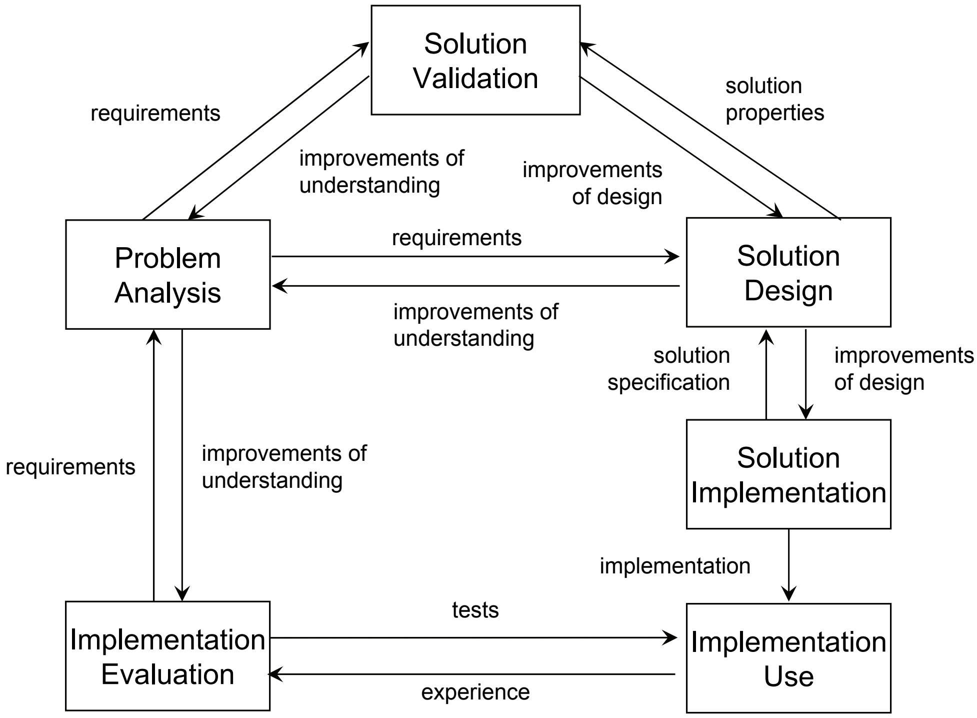


Figure 1.3: Activities in the engineering cycle including an implementation use activity. *Boxes* represent activities, *arrows* represent impacts. There is no perfect sequential relationship between activities. Adapted from [Wieringa et al., 2005].

Engineering Cycle. The engineering cycle in Figure 1.3 is basically a rational decision making structure [Wieringa et al., 2005]. It represents a list of activities that include problem analysis, solution design, solution validation, solution implementation, and implementation evaluation[2]:

- *Problem Analysis.* Examining a problem and defining the goals.
- *Solution Design.* Specifying one or more possible solution.

[2]Note that the activity Implementation Use is not a part of the engineering cycle, according to [Wieringa et al., 2005]

- *Solution Validation.* Analyzing the properties of specific solutions, and evaluating their potential to solve problems and achieve the desired goals.
- *Solution Implementation.* Implementation of a selected solution.
- *Implementation Evaluation.* Analysing the properties of the implemented solution. This might be a problem analysis task in another engineering cycle.

The important characteristic of the engineering cycle is that it describes a list of activities, but does not prescribe their sequential execution. One can start with problem analysis, with solution design, or even with validation. This approach is based on empirical findings that show that a human tends to perform these research activities in parallel. For example, as shown by [Cross, 2001] experienced designers develop their understanding of a problem parallel to designing a solution and validating its properties.

The engineering cycle approach contrasts with the view taken in the business science literature on the design science research methodology [March and Smith, 1995], [Hevner et al., 2004]. Firstly, the engineering cycle describes a more detailed set of activities needed to effect a design, compared to the two activities *build* and *evaluate* described in design science literature. Secondly, it does not require the execution of all activities, but emphasizes the non-sequential nature of design, which has been proven by many (see [Wieringa et al., 2005], [Akkermans and Gordijn, 2006] for a more detailed discussion on this issue). Thirdly, it distinguishes between validation and evaluation, which the design science literature does not [March and Smith, 1995], [Hevner et al., 2004]. Validation research is covered by *solution validation*, while evaluation research includes either *problem investigation* or *implementation evaluation*.

The difference between validation and evaluation is that in the former, a technique not yet implemented in practice is investigated, whereas in the latter, a technique already implemented in practice is investigated. Various research methods can be used for evaluation, such as field research, case studies etc. However, doing field research for validation is not a good idea, since the properties of a technique that has not yet been implemented are under investigation. Useful research methods for validation are mathematical analysis, formal proofs or laboratory experimentation.

Validation is an essential part of the design process. For example, when civil engineers come up with a new design for a bridge, they do not immediately go and test it in practice, as that may have severe consequences. The usual way of working is to scrutinize the design by testing it in a computer-simulated environment and to rely on thorough mathematical proofs.

1.8 Research activities

The research activities carried out for this study are shown in Figure 1.4. This figure is based on the engineering cycle and also indicates which chapters in this thesis address certain activities.

Following the guidelines laid down in the engineering cycle, the research activities were not performed in a sequential order. We started with problem analysis and moved on to solution

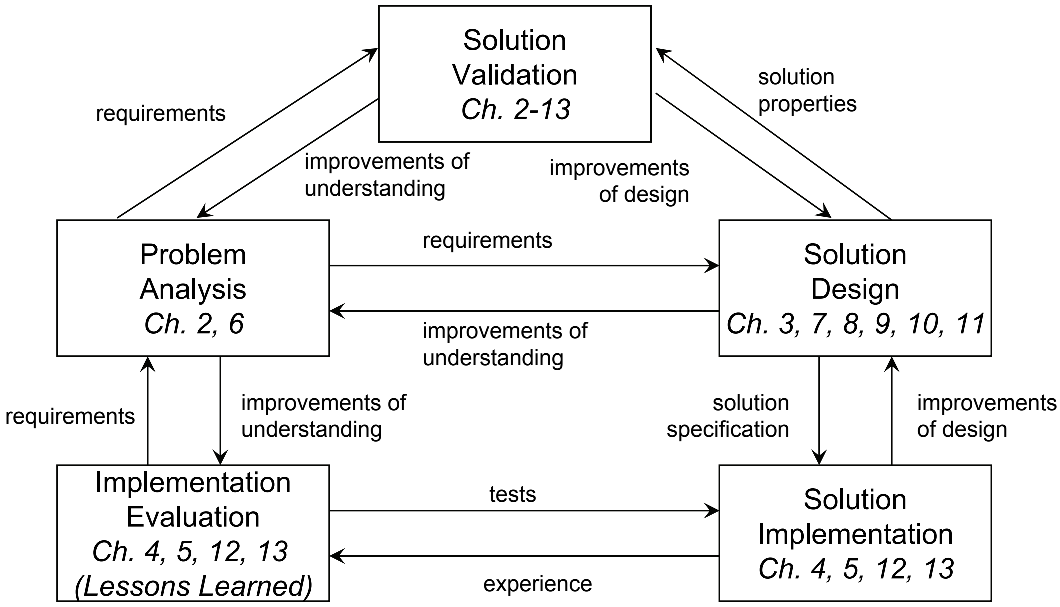


Figure 1.4: The research process

design, implementation and evaluation. However, there were multiple iterations of all activities. Deeper insight into each activity was gained by undertaking other activities, often in parallel. In this thesis we report on the results of this complex design process.

In this section, we describe each research activity of the engineering cycle, how they were carried out, and finally, we highlight the important results.

1.8.1 Problem Analysis

We examine the problems pertaining to the design of controls in networks. This step is described in detail in Chapter 2. Firstly, we define the requirements for our design methodology, then, we make an extensive literature review of the existing methodologies and theories of controls. The goal of this literature study is to find out if off-the-shelf solutions exist for designing the inter-organizational controls that meet our requirements. Unfortunately, as we will show, no such methodology is available. Although methodologies on design of control mechanisms are available in the internal control literature, they do not satisfy our requirements, because they are not described from the economic value perspective or from the network perspective. Therefore, the literature review in Chapter 2 is used as input for the development of the e^3*control* approach.

1.8.2 Solution Design

The solution design activity includes the development of the e^3*control* methodology which, as already explained, the e^3*control* methodology consists of three parts: the e^3*control* ontology, the e^3*control* patterns and the e^3*control* design framework.

We start with the e^3*control* ontology, described in Chapter 3. The e^3*control* ontology is an extension of the e^3*value* ontology, adapted to model control problems and control mechanisms. The development of the e^3*control* ontology is performed in iteration with its implementation in two case studies in the music industry (see Chapter 4) and in the health care industry (see Chapter 5).

We also develop e^3*control* patterns, which are described in Chapters 6 to 10. For their development, we rely on the methodological guidelines taken from patterns literature, as described in Chapter 6. According to the relevant literature, patterns have to represent proven solutions for recurrent problems and they have to describe *existing* knowledge. The patterns are elicited based on knowledge taken from both theories as well as case studies.

1.8.3 Solution Implementation

Empirical methods should be used to evaluate the results of the solution design [Wieringa et al., 2005]. Several research strategies are available, e.g. experiments, surveys, and case studies. We choose a *case study* as a method of evaluating the e^3*control* approach.

A case study is an 'empirical inquiry that investigates a contemporary phenomenon within its real-life context' [Yin, 1994]. [Yin, 1994] has identified some specific types of case studies: *exploratory*, *explanatory*, and *descriptive*. A more suitable categorization for our research is to distinguish between the *extracting* and the *developing* case-studies [van Aken, 2004]. The extracting case-study is a kind of best-practice research and is aimed at uncovering artifacts (e.g. technological rules) already used in practice. In the developing case-study the artifacts are developed and tested by the researcher(s) in close collaboration with people in the field. The nature of the case studies reported in this thesis is *developing*. The goal of our case studies is to test how the e^3*control* is able to handle real-life problems.

Due to the nature of the engineering cycle, the development of the e^3*control* approach is carried out parallel to the case studies. During the development stage, the e^3*control* approach is applied to design the case studies and is changed many times as a result of feedback, which is also provided the other way around. As a result, the e^3*control* models describing the case have to be changed or additional data collected. In this thesis we report only on the final outcome of this design process.

Case studies We perform four case studies for this research, each of which will now be described in short. The first two case studies involve the evaluation of the e^3*control* ontology and the initial e^3*control* design framework. The last two case studies involve the evaluation of the e^3*control* patterns and the e^3*control* design framework, adapted for the use of the patterns.

- **Internet Radio.** This case study focuses on the clearance of rights for broadcasting music content via the Internet. In order to broadcast music commercially, a radio station has to obtain several licenses and pay for several copyrights, including a *right to broadcast* music to the public. Currently, there are no reliable procedures to identify how many tracks are broadcasted, what tracks are broadcasted and how many listeners listened to which track. When reported by the radio stations, this information is not reliable because they want to reduce the royalty payments and may therefore report the information incorrectly. In this case study, we use e^3*control* to design a reliable procedure for controlling the broadcasting information.

- **Dutch health care services.** A study is made into the complicated payment process within the Dutch health care system for the treatment of long-term and chronic diseases. In The Netherlands, such diseases covered by public insurance which is paid for by the government but involves many controls. The goal of this case study is to design existing controls and to analyze the underlying control problems. In fact, we perform a sort of reverse engineering of the past and present procedures in the health care system to see if e^3*control* could describe it. In addition, we design a future situation, which has not yet been implemented in practice. A remarkable feature of this study is that the health care system is a highly regulated sector, in which production, distribution, and pricing decisions are made by a regulatory authority rather than determined by market forces. The specific question examined in this case study is whether the e^3*control* approach, which relies on e^3*value* and its notion of the reciprocal economic exchange, makes sense in such a non-profit setting.

- **Renewable Electricity Certificates.** This study looks into the system for stimulating the generation of electricity from renewable sources, such as wind turbines and solar panels. The case specifically focuses on such a system which has been implemented in the UK, where the Renewable Obligation (RO) law places an obligation on electricity suppliers, who are licensed to supply electricity in the UK, to source a certain proportion of electricity from renewable sources [OFGEM, 2004b]. Suppliers meet their obligations by presenting Renewable Obligation Certificates (ROCs) which correspond to a certain amount of the electricity bought by the supplier. In addition, ROCs can also be traded separately from the electricity supply, e.g. a supplier who had bought insufficient renewable electricity can buy more ROCs to comply with the regulation. The focus of this case study is whether the e^3*control* patterns could describe the mechanisms behind the ROC scheme. Unlike in the Internet radio case, the mechanisms here had already been implemented.

- **Beer Export.** The case concerns an excise collection procedure within the EU. When excise goods like beer and cigarettes are sold, the seller must pay a special tax called *excise*. If the goods are sold within the EU, the seller does not need to pay the excises in the country of export; the excises are paid in the country of import. However, the collection of excises poses many control problems. For example, exporters may *overstate* the amount of exported excised goods to avoid excise payments. This happens because of control weaknesses in the existing excise reporting procedure. The goal of this case study is two-fold, firstly, we undertake a kind of re-engineering of the existing procedure and, by using the e^3*control* patterns, suggest how it could be improved. The second goal of the case study is to suggest a scenario to replace the existing paper-based control procedure using new smart seal-based technology. The issues examined in this case study are (1) whether the e^3*control* patterns are able to describe weaknesses in the existing procedure and (2) whether they are able to design a new smart seal-based solution of the procedure.

The case studies were selected primarily by (1) variety of business sectors and (2) availability. The availability of the cases depended on the projects that provided access to domain experts and case material, and on the willingness of the domain experts and companies to cooperate. Hence, (a) generalizability of case observations is limited, and (b) completeness of control mechanisms is not guaranteed by these cases. These and other issues of research design validity are discussed later in this section.

1.8.4 Implementation Evaluation

During implementation evaluation, we investigate the use of the e^3*control* methodology in practice. Techniques may not work as expected, so in practice there are usually multiple iterations between the solution design, the solution implementation and the implementation evaluation tasks.

The final evaluation of the research results is done by reflecting on the results and stating the **Lessons Learned** at the end of each case study. The lessons learned reflect mainly on the arguments given in support of the value modelling at the beginning of the thesis. These arguments are:

1. Controls are commercial services with added value, which can be modelled from the value viewpoint;
2. Control instruments have inherent value aspects, which can be modelled from the value viewpoint;
3. Value viewpoint is an abstract rationale for procedural aspects of controls.

Furthermore, in the case of Music and Health care we discuss the ability of the e^3*control* ontology to describe the control problems and mechanisms in the case study. In the ROC and Beer case studies we also discuss the ability of control patterns to describe control problems and mechanisms. In particular, we are interested if the e^3*control* ontology and e^3*control* patterns provide sufficient analytical instruments to describe the existing controls or design new controls.

1.8.5 Solution validation

Two points can be addresses with respect to solution validation: (1) validation of the e^3*control* ontology and e^3*control* patterns and (2) validation of the case study models, which are built based on the e^3*control* or e^3*value* ontologies. The e^3*control* ontology was built using the principles of building ontologies from software engineering [Noy and McGuinness, 2001]. We have not looked into the validation of the e^3*control* ontology itself. However, the e^3*value* ontology, on which the e^3*control* is based, has been validated and was build using the ontology engineering principles [Gordijn, 2002]. The compliance of the the e^3*control* ontology with the ontology building principles can be proven by the fact that the e^3*value* software tool is able to represent the e^3*control* models.

The e^3*control* patterns are designed according to the rules of elicitation taken from the patterns literature. That is to say, each pattern is encoded into the *context-problem-solution* structure, an example of each pattern is provided, and the patterns vocabulary is defined. For more on this see Chapter 6.

With respect to validation of the case study models, we have observed that they comply with the restrictions of the e^3*control* or e^3*value* ontologies. For example, as in e^3*value* , the e^3*control* ontology prevents loop forming in the dependency paths. Furthermore, the one-way value transfers are only allowed in subideal e^3*control* models, but not in ideal e^3*value* models. So, we observe that there is no one-way value transfers in the ideal models and that the dependency paths do not contain loops. See chapter 3 on more details about the e^3*control* ontology, e^3*value* ontology and ideal and subideal value models.

1.9 Research design criteria

Four well-known tests can be carried out to judge the quality of empirical research. Construct validity, internal validity, external validity and reliability.

Construct validity. Construct validity means establishing the correct operational measures for the concepts being studied. This is especially problematic in case study research and has been a source of criticism because of potential investigator subjectivity. [Yin, 1994] proposed three remedies to counteract this:

- Use multiple sources of evidence;
- Establish a chain of evidence (A chain of evidence should be maintained to increase the reliability of the study. The procedure involves an external observer who follows the derivation of evidence from initial research questions to ultimate case study conclusions);
- Have a draft case study report reviewed by key informants.

In our case studies, we comply with all three requirements a discussion of which now follows. For more details on the data collection for each particular case, see the section Research Context of each case.

- **Internet Radio.** In this case, we reused the case *description* from previous research, e.g. [Pedrinaci et al., 2005]. The case description was composed by other researchers, namely by J. Gordijn, in cooperation with SENA[3] and within the realm of the OBELIX research project[4]. The author of the thesis has elicited knowledge by closely cooperating with J. Gordijn as well as by reading the research reports of the OBELIX project. See more on the research context in Chapter 4.

[3] In Dutch: De Stichting ter Exploitatie van Naburige Rechten. See www.sena.nl

[4] http://obelix.e3value.com

As a chain of evidence, the case study report, as presented in Chapter 4, refers to the original source of data where appropriate. Verification of the case study description as well as its final results was given by J. Gordijn who has a thorough knowledge of the domain. The description and results were also verified with SENA by J.Gordijn. The final information system model was verified with a security expert.

- **Dutch health care services.** This case study was conducted within the Freeband User eXperience (FRUX) Freeband project [Droes et al., 2005]. The data for the case study was collected by the author of this thesis and another researcher J. Hulstijn from (1) a series of semi-structured interviews with five experts from different health care organizations, (2) publicly available documents and research articles on the Dutch health care system and (3) government web sites. In addition, the author of the thesis also participated in regular project meetings, where relevant issues were discussed by the domain experts. See more on the research context in Chapter 5.

  As a chain of evidence, the case study report, as presented in Chapter 5, refers to the original source of data where appropriate. More references can be found in the two reports of the FRUX project [Kartseva et al., 2005c] and [Kartseva et al., 2006d]. The interviews with domain experts have been written and are available upon request.

  Verification of the case study description and its final results was given by presenting the e^3*control* models to two domain experts. The models were adapted to incorporate the feedback.

- **Renewable Electricity Certificates.** The data collection was carried out within the EU-funded research project BUSMOD[5] before this research started. During this project, data was collected by several researchers, including the author of this thesis. This mostly took the form of organized brainstorming sessions with domain experts from different organizations in the network. Some data, mostly concerning processes behind the ROC scheme, were collected from on-line public sources. See more on the research context in Chapter 12.

  As a chain of evidence, the case study report, as presented in Chapter 12, refers to the original source of data where appropriate. More references can be found in the deliverables of the BUSMOD project, which are available from the project web-site.

  Verification of the case study value models was given by the domain experts – participants of the BUSMOD project, in particular in workshop sessions organized during the project. The process models were not verified with domain experts, but only verified against literature sources and with other researchers.

- **Beer Export.** This case study was carried out within the EU-funded research project ITAIDE[6]. During this project, the data on how the excise system operates were collected mainly by several other researchers (Z. Baida, Y.-H. Tan, B. Rukanova and J. Liu) during both interviews and organized brainstorming sessions with stakeholders from different organizations in the network. The author of the thesis elicited knowledge by closely

[5] http://www.e3value.com/projects/ourprojects/busmod/
[6] www.itaide.org

cooperating with these researchers, by reading their project reports and by participating in several sessions with the domain experts. In addition, one extensive interview, conducted by the author of the thesis with two domain experts, was specifically targeted to elicit knowledge about the control problems within the case. See more on the research context in Chapter 13.

As a chain of evidence, the case study report, as presented in Chapter 13, refers to the original source of data where appropriate. More references can be found in the deliverables of the ITAIDE project, which are available from the project web-site. All interviews with the domain experts were tape-recorded.

Verification of the case study value and process models was given by the domain experts – participants of the ITAIDE project, in particular in brainstorming sessions organized during the project as well as during an interview in a personal meeting.

Reliability. The objective of reliability testing is to demonstrate that the results of a case study can be the same if a second investigator follows exactly the same procedures described by an earlier investigator and conducted in the same case study. We believe that the procedures we carry out for data collection and chain of evidence ensure reliability. It is further ensured because the analysis is done using rigorous and well-formalized conceptual modelling techniques, such as e^3*value* , e^3*control* and UML activity diagrams.

Internal validity. Internal validity means proving that reasoning and conclusions are correct in the context of a case study. To ensure this, we can compare the results that we get from the e^3*control* modelling with the knowledge and conclusions drawn by domain experts or provided by literature. As already mentioned in the Construct Validity section, a validation of the results with the domain experts is performed to some extent in each case study.

External validity. External validity is the degree to which we can generalize the findings of case studies and is usually a major barrier for the case study research. Obviously we cannot make any sound claims for the generalizability of e^3*control* approach and e^3*control* patterns based on four case studies. The generalizability claim cannot be made, since (1) we cannot prove that the set of patterns is complete (most likely there are still control mechanisms missing) and (2) we cannot prove that the patterns give the right solution for all possible cases.

However, we undertake several measures to ensure at least some generalizability. Firstly, the e^3*control* methodology in general, and e^3*control* patterns in particular, is mostly based on the internal control theory, the background of which reflects decades of practice and academic research. This helps to ensure that we will capture the most important aspects of the design of controls.

Secondly, the findings are tested in four large case studies, which cover various sectors. If the e^3*control* patterns are found in these various sectors, this to some extent will give the impression that the patterns are not domain-dependent.

1.10 Research contribution

The design of control mechanisms for network organizations is uncharted territory. The e^3*control* approach contributes an integrated perspective to controls design and structures existing knowledge on controls by employing (1) the *economic value viewpoint*, (2) the *network perspective*, (3) the *patterns*.

The design of controls in networks is a multi-disciplinary task. It obviously involves economic and legal aspects, but computer science issues are also relevant (many controls are implemented in information systems), as well as inter-organizational business process design (many contracts stipulate how, and in which sequence, business transactions should be carried out, and by whom). Therefore, the e^3*control* approach offers an integrated perspective, which enables the design of both economic and procedural aspects of controls.

Although the process and information systems viewpoints on controls have been developed for internal controls (see e.g. [Romney and Steinbart, 2006, ISACF, 2005, COSO, 2004]), the integration with economic aspects of controls has been left out. As has been claimed earlier, the development of the economic value perspective on controls is very important. The e^3*control* approach facilitates the evaluation of the impact of control mechanisms on the business models that underlie the network organization. This is innovative compared to the field of internal control, where, conventionally, only process models are considered for designing controls, and no conceptual modelling tools are used to perform a cost-benefit analysis.

The second contribution is the *network perspective on controls*. In network organizations, the opportunistic behaviour of some participants may prevent other participants from achieving their strategic objectives, which can cause the network to fall apart. Therefore, in order to satisfy the objectives of each network participant, a more general approach to control is needed. The network perspective of e^3*control* , agreed upon and subsequently implemented as a standard by network participants, considers controls to migrate the opportunistic behaviour of the peer participants. This view contrasts to the traditional practice of the internal control theory (e.g. [Romney and Steinbart, 2006, Starreveld et al., 1994]), in which each organization within the network only manages the activities within its own boundaries and where no explicit difference is made between threats coming from inside or outside of the organization. The contribution of e^3*control* is not to invent new controls, but to restructure existing ones (i.e. internal controls) and describe them from the network perspective.

The use of requirements engineering techniques, such as conceptual modelling and multiple viewpoints, is innovative in the field of control design. Even though these techniques are familiar to the internal control field, their full potential is not being utilized. The purpose of e^3*control* is to assist in the design of inter-organizational controls in a systematic way. To facilitate the design process we propose setting up a library of e^3*control* control patterns to structure existing knowledge and make it transferable to the inter-organizational domain, which is a contribution in itself. Furthermore, such a structured way of design is different from standard practice in internal control, which involves work with databases of hundreds of very specific controls which are only applicable to one domain. Describing the controls by using only six general patterns is a contribution of our work.

1.11 Publications

Major parts of this thesis have been published in conferences and journals. They include:

1. Kartseva, V., Gordijn, J., and Tan, Y-H. (2007) Design of Networked Enterprises Using e3control Patterns, In Proceedings of the 15th IEEE International Requirements Engineering Conference, October, 2007, India, New Delhi.

2. Kartseva, V., Hulstijn, J., Tan, Y-H., Gordijn, J. (2007) Control Patterns in a Healthcare Network, In: Proceedings of the 8th Annual International Conference on Digital Government Research, May 2007, Philadelphia, PA.

3. Kartseva, V., Hulstijn, J., Gordijn, J. and Tan, Y.-H. (2006) Modelling Value-based Inter-Organizational Controls in Healthcare Regulations, Proceedings of the 6th IFIP conference on e-Commerce, e-Business, and e-Government (I3E 2006), Turku, Finland.

4. Kartseva, V., Hulstijn, J., Tan, Y.-H., Gordijn, J. (2006) Towards Value-based Design Patterns for Inter-Organizational Control, Proceedings of the 19th Bled Electronic Commerce Conference, Bled, Slovenia.

5. Kartseva, Gordijn, J. and Tan, Y.-H. (2006) Inter-Organisational Controls as Value Objects in Network Organisations. In Eric Dubois and Klaus Pohl editors, Proceedings of The 18th International Conference on Advanced Information Systems Engineering, Vol. 4001:336-350 of LNCS, Springer, Berlin.

6. Kartseva, Gordijn, J. and Tan, Y.-H. (2005) Towards a Modelling Tool for Designing Control Mechanisms in Network Organisations, International Journal of Electronic Commerce (M.E. Sharpe), Winter 2005-6, Vol. 10, No. 2, pp. 57-84.

7. Kartseva, V., Gordijn, J. and Tan, Y.-H. (2005) Designing Control Mechanisms for Value Webs: The Internet Radio Case Study, Proceedings of the 18th BLED conference (e-Integration in Action), D. R. Vogel, P. Walden, J. Gricar, G. Lenart (eds.), Bled, Slovenia.

8. Kartseva, V. and Tan, Y.-H. (2005) Towards a Typology for Designing Inter-Organisational Controls in Network Organisations, Proceedings of the 38 the Annual Hawaii International Conference on System Sciences, R.H. Sprague, (ed.), IEEE Computer Society Press, Big Island, HI, USA.

9. Kartseva V. and Y.-H. Tan "Designing Control Mechanisms for Value Exchanges in Network Organisations", in: Formal Modelling for Electronic Commerce: Representation, Inference, and Strategic Interaction, Steven O. Kimbrough & D. J. Wu (eds.), pp. 231-244, Springer-Verlag, Berlin.

10. Kartseva, V., Gordijn, J. and Y.-H. Tan, "Analysing preventative and detective control mechanisms in international trade using value modelling", Proceedings of the 6th International Conference on Electronic Commerce, Marijn Janssen, Henk G. Sol and Rene W. Wagenaaar (eds.), Delft, The Netherlands, ACM, pp 51-18, 2004.

11. Kartseva, V., Tan, Y.-H. and Gordijn, J. (2004) Developing a Modelling Tool for Designing Control Mechanisms in Network Organisations, Proceedings of the 17th Bled International e-Commerce Conference, Bled, Slovenia.

12. Kartseva, V. Gordijn, J. , Tan, Y.-H. (2004) Value-based Business Modelling for Network Organizations: Lessons Learned From the Electricity Sector, Proceedings of the 12th European Conference on Information Systems, Turku, Finland.

13. Kartseva, V., Gordijn, J., and Akkermans, H. (2003) A Design Perspective on Networked Business Models: A Study of Distributed Generation in the Power Industry Sector, Proceedings of 16th Bled conference, pp. 434-445.

1.12 Outline of the research

This thesis is structured as follows. Chapter 2 elaborates on the theoretical foundation of this research and consists of business control theories and value modelling. This chapter argues why an economic value perspective is important when considering controls. In addition, it provides the reader with a broad literature review of the field of internal and inter-organizational controls.

In Chapter 3 we develop the e^3*control* ontology and the e^3*control* design framework. In addition, we discuss the design of rights and evidence documents in e^3*value* , which are typically present in controls. In Chapters 4 and 5 we discuss the e^3*control* approach by presenting two case studies: the music industry and health care sector. We also present the lessons learned.

Chapters 6, 7, 8, 9 and 10 deal with the development of the e^3*control* patterns which represent accepted design knowledge of inter-organizational controls. We start with the elicitation of these patterns in Chapter 6. Furthermore, we develop a vocabulary to describe the patterns in Chapter 7. We describe the actual control patterns in Chapters 8 and 9, and deal with the delegation of control activities in Chapter 10.

Chapter 11 shows how to use the patterns for a specific case at hand. Chapter 12 and 13 discuss two case studies in the electricity and international trade sectors to demonstrate and evaluate the application of the patterns. Finally, the thesis ends with the conclusions chapter, in which we come back to the research objective of this thesis and present our conclusions.

Chapter 2

Designing controls in network organizations

The goal of this chapter is two-fold. Firstly, we define the requirements for our design methodology, then we undertake an extensive literature review of the existing methodologies and theories of controls. As a result, we argue in favour (1) of developing a value-based design methodology for inter-organizational controls and (2) of using a conceptual modelling approach to design controls. Finally, we suggest developing *patterns* of inter-organizational controls to represent design knowledge explicitly. We start with reviewing existing theories on controls.

2.1 Inter-organizational controls

In this section we review theories taken from literature which consider controls in business organizations or networks. We see a network organization as a set of actors (e.g. companies or individuals) who have direct or indirect business relationships with each other. The network actors have to comply with certain *obligations*, imposed either by contracts between the actors or by law or regulation, e.g. European Union environmental laws that apply to all companies operating within the EU. Some actors may behave *opportunistically* with respect to these obligations and not comply with them.

Inter-organizational controls are the measures implemented by the network participants, jointly or individually, to mitigate such opportunistic behaviour. More precisely, we may define inter-organizational controls as *measures one or more party can use to prevent, detect or correct opportunistic behaviour of its counter parties*. This definition implies the network perspective on controls, described in Chapter 1.

Before proceeding with inter-organizational controls, we present a review of the internal control theory [Hollander et al., 1999], [Romney and Steinbart, 2006]. Although the internal control theory focuses mainly on controls inside organizations, we consider it to be a good starting point for understanding the concept of control in general. Furthermore, we consider *agency theory* which renders a useful introduction to inter-organizational controls. Finally, we consider the work on inter-organizational controls taken from a series of publications of [Bons, 1997], [Bons et al., 1998], which provides a starting point in the design of inter-organizational controls.

2.1.1 Internal control theory

The internal control theory views an organization as a set of *business processes* which are a series of *operational activities* intended to accomplish the strategic objectives of an organization [Hollander et al., 1999]. The operational activities are also called *events* [Hollander et al., 1999]. Traditionally, several standard business processes known as *transaction cycles* are identified within a hierarchical organization.

Transaction Cycles and Activities

Regardless of the type of goods and services it provids, each organization has at least three business processes, also called *transaction cycles* [Romney and Steinbart, 2006]. The main cycles are the *Expenditure Cycle*, the *Conversion Cycle*, and the *Revenue Cycle*.

The *Expenditure Cycle* incorporates only those activities and objects needed for acquiring, maintaining and paying for resources required by an organization. These activities are ‘request goods or services’, ‘order goods or services’, ‘receive goods or services’, ‘store and maintain goods or services’, and ‘pay for goods or services’. The *Revenue Cycle* consists of activities and objects needed for selling and delivering goods or services to customers and the collecting of payments. They are known as ‘receive an order for goods or services’, ‘select and inspect goods or services to be delivered’, ‘prepare goods or services to be delivered’, ‘deliver goods or services’, and ‘receive payment for goods or services’. The *Conversion Cycle* consists of activities that convert the acquired resources into goods and services for customers, e.g. assembling, growing, manufacturing, transporting, distributing, educating.

Internal versus external activities. Every activity in a transaction cycle can be classified as internal or external [Bons, 1997]. *Internal activities* are entirely within a party and are not visible to other parties. *External activities* are those activities that directly involve - and are visible to - outside parties. Expenditure and revenue cycles incorporate both internal and external activities. The activities ‘request goods or services’, ‘store and maintain goods or services’, ‘select and inspect goods or services to be delivered’, ‘prepare goods or services to be delivered’ are all internal. External activities are ‘order goods or services’, ‘receive goods or services’, ‘pay for goods or services’ in the expenditure cycle and ‘receive an order for goods or services’, ‘deliver goods or services’, and ‘receive payment for goods or services’ in the revenue cycle. The conversion cycle comprises internal activities only. In the realm of the network perspective, we only focus solely on the external activities of network participants.

Control problems

Every organization faces a multitude of risks which can be defined as exposure to the chance of injury or loss [Hollander et al., 1999], p.189. A potential risk threatens some aspects of the organization's operation or even its very existence. An organization faces many types of risks, namely strategic risk, decision risk, operating risks, information risks and financial risks [Hollander et al., 1999], p.191.

We focus primarily on *operating risks*, which are those related to the incorrect execution of activities of transaction cycles. According to the internal control theory, these risks lead to a loss of financial resources in the organization. Some examples of common risks include theft of cash by an employee, paying for the same goods twice, and accepting unordered goods.

Each activity in the transaction cycles has associated risks. [Hollander et al., 1999] suggest using a *REAL* model as a guide to recognizing these risks for each activity. *REAL* is an acronym for ***R**esource, **E**vent, **A**gent and **L**ocation* and is an extension of the *REA* model suggested by [McCarthy, 1982] for the design of accounting information systems. This model shows that every operating activity can result in unintentional errors and intentional irregularities with one or more of the following characteristics:

- An activity takes place at the wrong time or in the wrong sequence (or does not take place at all)
- An activity is carried out without the proper authorization
- An activity which involves the wrong internal agent
- An activity which involves the wrong external agent
- An activity which involves the wrong resource
- An activity which involves the wrong amount or number of of resources
- An activity carried out at the wrong location

According to this classification, a shipment activity may involve such risks as late shipment of goods, shipment without proper authorization, shipment to the wrong buyer, shipment of unordered goods, shipment of the wrong quantity of goods, and shipment of goods to a wrong location.

Control mechanisms

Internal control literature describes many different controls[1], specific to the risks of every transaction cycle. We are not going to list them here as they can be found in other sources such as [Romney and Steinbart, 2006] and [Hollander et al., 1999]. An example of such a control for a Shipping activity in the Revenue Cycle is "Reconciliation of sales order with picking ticket and

[1]In the internal control literature the term 'controls' is being used to refer to both control activities and control mechanisms. A *control activity* is an operational activity performed to minimize or eliminate the risk (see [Hollander et al., 1999], p. 189). This can be confusing, since a control activity is a not the same as a control mechanism. A control activity is only a part of a control mechanism. A control mechanism (e.g. authorization) *consists of* control activities, but also of other measures, such as the order of the activities in a specific situation. In this thesis, the term 'control' refers only to control mechanisms and we refer to control activity explicitly.

packing slip". The related control problem is errors in shipping, e.g. shipping wrong merchandize.

Conventional descriptions of internal controls are very domain-specific. Firstly, separate risks and controls are described for every specific activity of each transaction cycle. Secondly, they assume the presence of some specific documents (e.g. the picking ticket in the example above) or parts of IS (e.g. a credit card processing system). As a result, an enormous number of risks and controls known to a control expert. [Romney and Steinbart, 2006] lists 12 common risks in the Revenue Cycle[2] and 17 in the Expenditure Cycle[3]. Each risk corresponds to around three-four control mechanisms. So, a control expert, who wants to design controls for the Expenditure Cycle and Revenue Cycle, should be familiar with about a hundred controls.

Internal control literature also offers classifications of controls, for instance, to their use: whether they are used to prevent, detect, or recover from errors and irregularities. The purpose of each control is evident from their name [Hollander et al., 1999]:

- *Preventative controls* focus on preventing an error or irregularity
- *Detective controls* focus on detecting when an error or irregularity has occurred
- *Corrective controls* focus on recovering, repairing the damage from, or minimizing the cost of an error or irregularity.

Using credit card processing systems to avoid cash theft by cash processing employees is an example of a preventative control. Demanding a comparison of the cash in the cash drawer with the total sales accumulated by the cash register during an employee's shift is an example of a detective control. An example of a corrective control is a policy of deducting the amount of a cash drawer shortage from an employee's pay. This policy also serves as a preventative control, since it discourages the theft.

Control principles

Another classification of controls distinguishes between segregation of duties, authorizations, access controls, proper documents and independent verifications [Romney and Steinbart, 2006][4]. Notice that there is an overlap with the categories of preventative, detective and corrective controls. For example, segregation of duties is both a detective and a preventive control.

Authorization. Every employee should have *proper authorization* to perform any action, so that if something goes wrong *accountability* could be established.

Segregation of duties. No employee should be assigned too much responsibility according to the following principles:

[2] See p. 381 of [Romney and Steinbart, 2006]

[3] See p. 435 of [Romney and Steinbart, 2006]

[4] [Hollander et al., 1999] use a different terminology, but the meaning remains the same

1. An employee who is responsible for custodial activities (e.g. handling cash, inventories, writing checks, receiving checks) should not be the same employee who is responsible for recording functions (e.g. maintaining journals, preparing reconciliations, performance reports). This prevents employees from falsifying records in order to conceal theft of assets entrusted to them.
2. An employee who is responsible for custodial activities should not be the same employee who is responsible for the authorization of these activities. This prevents authorization of a fictitious or inaccurate transaction as a means of concealing asset theft.
3. An employee who is responsible for recording activities should not be the same employee who is responsible for the authorization of these activities. This prevents an employee from falsifying records to cover up an inaccurate or false transaction that was inappropriately authorized.
4. An employee who is responsible for a verification (checking) activity should not be the same employee who is responsible for the activity being verified (checked). This prevents an employee from falsifying results of verification to cover up an inaccurate or false transaction.

Audit trail. Every activity should be recorded to create a proper *audit trail*. More specifically, according to the *REA* model [McCarthy, 1982], a change in a resource as well as the executing agent should be recorded for every event. For instance, in a purchasing activity, the goods purchased and the employee who purchased them should be recorded.

Adequate documents and records. All documents should comply with certain requirements. For example, every document has to have space for a signature and they also have to be sequentially pre-numbered so each can be accounted for and any missing document can be detected.

Access restrictions. Information and assets have to be protected and access to them should only be granted to authorized persons. This control is closely related to segregation of duties because the authorization is assigned based on the segregation of duties principles.

Independent verifications. Every activity in an organization has to be periodically checked, verified or reviewed. In accounting such verifications may concern accuracy and completeness of records and are carried out by reconciling the records with another set of records, or by reconciling them with a material reality. Other points of attention are how often verification should take place, should it be direct or indirect, what documents and which material reality provides ‘true’ information about the state of affairs, and how this ‘true’ information should be collected. This principle is closely related to the fourth segregation of duties principle, since a verification activity has to be performed by a person, who is independent of a person, performing the activity being verified.

Henceforth, we call these categories *control principles*, to distinguish them from the control mechanisms themselves. One difference between a control mechanism and a control principle is that the principles are the building blocks of the control mechanisms. For example, a

“Reconciliation of sales order with picking ticket and packing slip” mechanism requires the implementation of several control principles: the correct segregation of duties between a party that reconciles the order and the one who records the sales order, correct information placed on the picking ticket and packing slip, proper authorization of the party who carries out the reconciliation, etc. The control principles can be seen as the *design rules* of internal controls.

Internal control theory in practice

The internal control theory is rather *practice-oriented* and provides the tools for practitioners, i.e. controllers and auditors, to apply and audit controls in internal business processes [Romney and Steinbart, 2006]. Practitioners use such conceptual modelling techniques as flow charts and data flow diagrams to describe administrative processes. The focus of internal control literature is on how an organization should arrange an internal accounting information system and internal business processes in such a way that loss from the potential opportunistic behaviour of employees is minimized. Practitioners focus mostly on audit of existing systems rather than on their design. Internal control literature pays most attention to procedural aspects of controls, such as safeguards of assets, access restrictions, segregation of duties and documentary evidence.

2.1.2 Agency theory

The controls in the internal control theory cannot be applied in a straightforward way to the control problems considered in the network perspective. Internal controls aim to mitigate control problems by adjusting the activities within a company’s boundaries and do not take into consideration activities and relationships with counter parties. In order to apply the theory to the network setting, we first need to understand familiar inter-organizational control problems and mechanisms to mitigate them. To do this, we consider the agency theory (e.g. [Ross, 1973, Arrow, 1985], for a survey see [Eisenhardt, 1989]).

The agency theory focuses on a relationship between two parties - a principal and an agent - in which a *principal* delegates an activity to an *agent*. Agency theorists do not distinguish between firms and markets in terms of how cooperation is achieved and in how firms and markets operate and produce coordination [Jensen and Meckling, 1976]. So, the *principal-agent framework* is also applicable in describing inter-organizational control problems and mechanisms. The theory argues that an agent behaves *opportunistically* under the conditions of (1) the principal and the agent being utility maximizers with bounded rationality, (2) uncertainty and (3) information asymmetry in favour of the agent.

The central dilemma examined by agency theory researchers is how to get the agent to act in the best interests of the principal when the agent has an information advantage over the principal and divergent goals or interests. Empirical evidence provided by the agency theory shows that outcome-based contracts align goals between principals and agents, that monitoring reduces self-serving behaviour of agents, that outcome-based contracts make agents more risk averse, and that behaviour-based contracts do the opposite [Eisenhardt, 1989]. Note that in this research, it is not our intention to test the assumptions of the agency theory empirically. We only use the principal-agent framework as a way of grasping and organizing inter-organizational controls.

Control problems in the agency theory

There are two types of control problems in the agency theory: hidden information and hidden action. *Hidden information*, also referred to as adverse selection, occurs when information possessed by the agent is concealed from everyone else, but the welfare of the principal depends on that information [Campbell, 2006]. For example, a producer may hide essential information about the quality of his product, as a result of which the buyer may make wrong decisions, e.g. by buying a product of a lower quality than desired.

Hidden information involves *hidden characteristics* and *hidden intentions* [Keil, 2005]. Hidden characteristics refer to the abilities and skills of the agent that are not 'common knowledge'. Hidden intentions refer to the goals and intentions of the agent not known by the principal.

The *hidden action* problem is also referred to as moral hazard. It arises when a principal hires an agent to carry out a task, but cannot be sure whether the agent does his work according to the contract [Campbell, 2006]. For example, opportunistic behaviour may lead an employee to overstate working hours if an employer does not know how long the employee spends on some particular piece of work.

The hidden information problem refers to opportunistic behaviour of an agent that arises *ex-ante*, so before a contract between a principal and an agent is signed, which means that this problem can be mitigated *ex-ante* as well. The hidden action problem refers to the opportunistic behaviour of the agent that arises *ex-post*, so after the contract between the principal and agent is signed. In fact, hidden information is an additional condition to the problem of hidden action. If not stopped in an ex-ante period, an agent who behaves opportunistically will cause problems in an ex-post period.

Control mechanisms in the agency theory

Agency literature describes several particular controls to mitigate the problems described above. The controls against the hidden information problem are *screening* and *signalling controls*; the controls against the hidden action problem are *monitoring* and *incentives* [Keil, 2005]. Here is a short description of each control.

Screening. Screening is the evaluation of the past activities of a potential agent by the principal. Such an evaluation satisfies certain criteria and enables the principal to employ an agent. For example, some companies screen employees by contacting their former employers before hiring them.

Signalling. The signalling mechanism was introduced by [Spence, 1973] and is based on the idea that an agent performs some action which credibly signals the information about the agent. Unlike in screening, a principal does not evaluate the past activities of a potential agent, but rather other information that provides a picture about an agent's abilities. There are two types of signalling information: indexes and signals [Keil, 2005]. Indexes involve information and data about the track record, economics performance, number of employees, etc. For example, in some cases a principal may decide that a large, well-known company is more suitable for certain work than a less known middle-size company. The second type of signalling information is *signals*, which are guarantees or certificates given by trusted parties. For instance, a company may prefer job candidates with a university degree, since that signals a certain level of knowledge of the candidate.

Monitoring. After a contract is settled, the principal can monitor the behavioural activities of the agent. A disadvantage of *behaviour monitoring* is that it can be costly. For instance, the principal might not have the expertise to judge the agent's behaviour correctly or he might not have access to the agent's activities, e.g. in overseas trade. In such cases the principal would need to hire a third party that has the necessary skills or access.

The principal can also abandon monitoring behaviour, and monitor the output instead. The crucial point of *output monitoring* is to formulate rules and restrictions in the agent's behaviour so that the output can be controlled unambiguously. For example, in outsourcing software development the principal might require the agent to follow coding and testing guidelines, functional specification, etc. [Keil, 2005]. The output can then be assessed against these guidelines.

Incentives. The implementation of monitoring mechanisms is often difficult and costly and can sometimes even be impossible. Another way to prevent an agent's opportunistic behaviour is to align his or her interests with those of the principal by means of an *incentive system*. Incentives motivate the agent to act in one way rather than another, given a choice of actions. A classic example is to make sure that a CEO's salary depends on a company's performance [Jensen and Meckling, 1976].

There are two types of incentives: *positive* and *negative incentives*. Positive incentives are rewards, negative incentives are punishments. Henceforth, we will call positive incentives just *incentives* and negative incentives *penalties*.

Incentives and penalties always require monitoring mechanisms to measure when the reward or penalty has to be given or not. In the example of the CEO, such a measure can mean an increase in profit at the end of the year.

2.1.3 Inter-organizational control theory

We have reviewed two control theories in business science: the internal control theory and the agency theory. On the one hand, the internal control theory provides a detailed practice-oriented approach to design controls. The control principles presented in this section can be seen as prescriptive design rules. However, these principles have been developed to be applied to internal controls. In fact, it is not that clear whether they make sense in the inter-organizational setting.

On the other hand, the agency theory provides a clear structure and description of common inter-organizational control problems and mechanisms. However, it does not provide detailed control principles behind controls in a way that the internal control theory does. The agency theory researchers do not focus on design details, but on the selection of appropriate controls and on empirical examination of performance of controls [Eisenhardt, 1989].

We need to merge both theories in order to design inter-organizational controls. On the one hand, we need to design control mechanisms to solve problems relevant to the network setting, as described in the agency theory. On the other hand, we would like to design them at the same level of detail as those described by the internal control principles.

In this section we make a review of existing work on inter-organizational controls, gained from a series of publications of [Bons, 1997], [Bons et al., 1998] and [Bons et al., 2000]. This work uses the principle-agent framework on the one hand and the internal controls on the other and describes control principles for several inter-organizational controls. It identifies two types of inter-organizational controls: those with a *preventative function* and those with an *evidence function*. While the preventative function covers the monitoring control of the agency theory, the evidence controls are, in fact, an inter-organizational interpretation of the audit trail concept of the internal control theory (see section 2.1.1).

An inter-organizational model, taken as a unit of analysis in Bons, has been borrowed from the agency theory. It assumes two actors who have a legal agreement to execute two operational activities. The principal agrees to execute a so-called *primary activity*, and an agent agrees to execute a *counter activity*. The *primary activity* is an "external, activity of a party that corresponds to a primary obligation in some underlying legal agreement". The primary activity is executed *in return for* a *counter activity*. Consequently, the principal is called the *primary actor*, and the agent is called the *counter actor*. As in the agency theory, the primary actor is uncertain about the counter actor and expects him to behave opportunistically.

Preventative function of controls

What Bons calls the controls with a 'preventative function' are in fact the **monitoring** controls of the agency theory. Unlike the agency theory, the Bons controls are described in a very detailed way so that clear guidelines for the procedural implementation of the controls can be derived. For example, they describes what operational activities have to be executed, in which order and by whom. Such level of detail is similar to the internal control theory.

[Bons, 1997] formulates three variations of controls with a preventative function[5]. The first mechanism takes into account that a principal monitors the counter activity of an agent by direct observation of this activity. The second and the third mechanisms cover the situation when observation is not possible; in this case, the principal has to rely on some other evidence provided by a trusted party; a trusted party is a party trusted by the primary actor to provide the correct evidence of the counter actor's actions.

Evidence function of controls

The evidence function of inter-organizational controls is to ensure that the party responsible for the performance of some activity receives evidence of execution of this activity[6]. [Bons, 1997]

[5] Here we refer to the general control principles II - IV, see [Bons, 1997], p.61-62

[6] Here we refer to the general control principles I and V, see [Bons, 1997], p.60-62

argues that since multiple autonomous parties are involved in networks, a situation in which someone behaves opportunistically usually leads to a (legal) dispute. In such cases, the function of inter-organizational controls is to establish evidence that may be used by both the principal and the agent in case of the dispute.

Evidence function controls are of two types: execution evidence and commitment evidence and they address different control problems. The execution evidence and commitment evidence controls are not described in the agency theory, as reviewed in section 2.1.2. However, they can be related to the *audit trail* principle of the internal control theory (see section 2.1.1). The audit trail requires every activity to be recorded so that accountability for actions can be established.

Execution Evidence. The execution evidence control problem is related to the counter actor's actions when accusing the primary actor of the wrong execution of the primary activity. The counter actor claims that the primary actor has not executed the primary activity properly, to which the primary actor does not agree.

To mitigate this problem, the counter actor must be able to testify about the completion of the primary activity by making use of some evidence document, which should be offered to the primary actor. In case of a dispute, the evidence document can be used by the primary actor as a proof of performance of the primary activity. An intuitive example of such an evidence document is a *receipt* in a shop. When a customer buys something in a shop, he must have a receipt. If, after a while, the shop accuses the customer of stealing goods, the customer can always show the receipt to prove that he paid for the goods.

Commitment Evidence. The commitment evidence control problem is related to the counter actor's denial of his commitment to the primary actor. For example, a seller may refuse to execute an order for a certain price by saying that that the agreement was for a higher price. A buyer may refuse to pay for goods by saying that that he did not order them. In the internal control theory, these problems result in uncollected cash and illegitimate orders in the revenue cycle ([Romney and Steinbart, 2006], p. 380).

To mitigate this problem, the commitment evidence control requires the counter actor to send the principal a confirmation of his commitment, e.g. in a form of an evidence document [Bons, 1997], [Weigand and de Moor, 2003]. This evidence can be used in a case of dispute. An intuitive example of the commitment evidence document is a contract. By signing a contract both parties create evidence of their commitment to each other.

The pre-execution requirement

An important aspect of the mechanisms described by [Bons, 1997] is a requirement to execute a primary actor's activity after a counter actor's activity. We call this a **pre-execution requirement**.

As controls of the preventive function, the principal has to obtain evidence of the counter activity *before* executing the primary activity[7]. Thus, *monitoring of the counter activity has to be done*

[7]see the general control principles II-IV [Bons, 1997], p.61-62

before the primary activity. A well-known example of such a requirement is a pre-payment agreement in trade. A seller can require payment upfront if the creditworthiness of a buyer is unsure.

The pre-execution requirement is also present in controls with the evidence function. As commitment evidence, the principal should receive confirmation of the agent's commitments *before* executing the primary activity[8]. As execution evidence, the principal should receive the confirmation of the primary activity *at the same* time as the primary activity is executed[9].

The pre-execution requirement is an important component of inter-organizational controls. In the internal control theory, the pre-execution principle is similar to the authorization principle, which requires an authorization to be issued before the execution of the activity being authorized (see section 2.1.1). In the pre-execution principle for monitoring, a positive outcome of the monitoring activity can be seen as an authorization to execute the primary activity.

Trust

A network organization can be interpreted as a number of binary transactions between actors. A lack of trust is likely between parties in a network who do not already have a business relationship. Trust has been defined as "The willingness of a party to be vulnerable to the actions of another party based on the expectation that the other party will perform a particular action important to the trustor, irrespective of the ability to monitor or control that other party"[Mayer et al., 1995]. Without prior trust, the party who invests in a transaction, called the *trustor*, is *uncertain* whether the other party, the *trustee*, will perform its part of the deal, or will defect and behave opportunistically. The uncertainty of the trustor about possible opportunistic behaviour also comes out in Gambetta's definition: "Trust is the subjective probability by which an individual A expects that another individual B performs a given action on which its welfare depends" [Gambetta, 1988]. However, trust does not have to depend on the trustor and trustee alone. Institutional control measures can be used to guarantee performance according to a contract. In general, the purpose of an inter-organizational control mechanism is to reduce the uncertainty of the trustor and provide enough guarantees for both parties to engage in a transaction [Tan and Thoen, 2000].

In Bons's work, trust has a direct relation to the concept of *uncertainty*. Therefore, little trust by the primary actor in the counter actor can be interpreted in the work of Bons as 'the primary actor is uncertain in the counter actor'.

[Bons, 1997] incorporates the concept of trust in his controls. A general guideline is that a control is needed in case the primary actor has little trust in the counter actor. The pre-execution requirement can be relaxed if the primary actor does trust the counter actor. For example, a pre-payment arrangement might not be required if a seller is certain that a buyer will pay. Such trust usually exists in long-lasting relationships and is absent in relationships limited to a one-time transaction.

[8] see the general control principles V [Bons, 1997], p.62
[9] see the general control principles I [Bons, 1997], p.60

2.1.4 Management control theory

Another trend in the literature on business controls is the management control theory [Anthony and Govindarajan, 2003], [Merchant and van der Stede, 2007] the scope of which is broader than that of the internal control theory. The objective of controls is seen not only as to mitigate opportunistic behaviour with procedural measures, as taken in this work, but also on designing controls in such a way as to motivate partners or employees to reach the objectives efficiently and effectively [Geringer and Hebert, 1989], [Groot and Merchant, 2000]. As a result, the management control theory considers controls more as *coordination mechanisms* to improve the overall performance of an organization or a network. Usual management controls include budgeting, reporting and performance incentives. The procedural controls constitute only a small part of the coordination mechanisms. The management control literature on inter-organizational relations focuses more on the second objective and less on procedural aspects of inter-organizational controls.

Although our focus is on procedural controls and not on coordination mechanisms, the management control theory provides several classifications of inter-organizational controls that are worth considering for a general understanding of a domain. We will now describe the classifications in more detail. Their application to the previously discussed controls can be seen in Figure 2.1.

Ex-ante and ex-post controls

Controls can be classified as ex-ante and ex-post. We call controls *ex-ante* if they are executed before a contract is settled, i.e. signed. Controls executed after the contract is settled are called *ex-post*.

The screening, signalling and commitment evidence controls detect and prevent an agent's opportunistic behaviour that occurs ex-ante, and they are therefore executed ex-ante (see section 2.1.2). The monitoring of contract performance and execution evidence controls detect and prevent the opportunistic behaviour that occurs ex-post and they are therefore executed ex-post (see section 2.1.2, 2.1.3). Incentives are executed both ex-ante and ex-post (see section 2.1.2). Settlement of contracts where incentives are defined (e.g. agreements on ownership structure) take place ex-ante. However, rewarding or punishing of the agents takes place ex-post.

In addition, controls can be assigned to phases of the transaction cycle [Weigand and de Moor, 2003], see also [Bons, 1997], p. 30. The process of concluding a transaction consists of four phases: the **preparation phase** and the **negotiation phase**, which occur ex-ante, and the **execution phase** and the **acceptance phase**, which occur ex-post. Therefore, screening and signalling controls are executed during the preparation phase. Commitment evidence controls are executed in the negotiation phase. Monitoring is executed during the execution phase. Execution evidence controls are executed in the acceptance phase. The contracts on incentives are settled in the negotiation phase, while rewards/penalties are paid in the execution phase.

Contractual and procedural controls

The management control literature distinguishes between contractual obligations, henceforth called *contractual controls*, and formal organizational mechanisms for cooperation, henceforth called *procedural controls* [Ouchi, 1979], [Sobrero and Schrader, 1998]. Incentives are contractual controls, since they involve contractual arrangements that settle financial incentives to align the interests of the principal and the agent. All other controls - screening/signaling and monitoring, and the evidence-related controls - are procedural, since they rely on procedures and do not involve financial incentives. In addition, incentives require presence of procedural monitoring controls (see section 2.1.2).

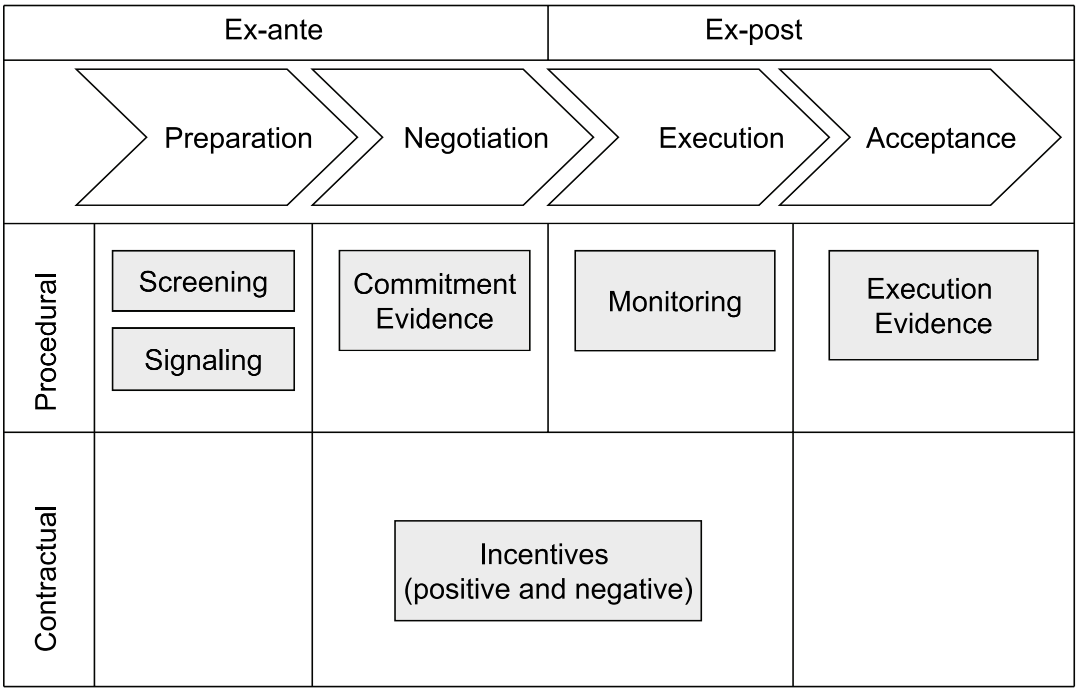


Figure 2.1: Types of formal controls

Formal and informal controls

Controls can be *formal* and *informal* [Ouchi, 1979]. All the controls discussed so far - screening, signalling, monitoring, evidence creation and incentives - are formal controls. Informal controls, also referred to as social controls, concern the informal cultures and systems which influence members and are essentially based on mechanisms inducing self-regulation e.g. through the creation of shared values and creation of trust [Ouchi, 1979], [Dyer and Singh, 1998], [Dekker, 2004].

Some formal controls also contribute to informal controls. For example, screening and signalling can also be seen as informal controls. The informal part of screening and signalling is that the selection of a good partner contributes to a creation of trust [Ouchi, 1979], [Dekker, 2004]. In this work, we focus on procedural aspects of the screening and signalling controls.

Informal controls are more a condition rather than a mechanism that can be designed. Trust and reputation either exist or not: The presence and absence of trust and reputation define to what extent other formal controls have to be implemented [Dekker, 2004]. A common assumption is that the more social control is present there in a relationship, the less formal control is required.

2.2 Design of controls

The objective of this study is *to develop a methodology that would support human analysts in designing controls against opportunistic behaviour in networks of organizations.* In the previous sections of this chapter we have reviewed the existing methodologies and theories on the design of controls. Before we draw any conclusions about whether any of these theories and methodologies suit our needs, we discuss what characteristics we would like our methodology to have.

Networks are characterized by collaborative decision making. Because companies participate in networks voluntarily, but seek self-interest, any agreements in networks, including controls, are not usually decided upon by an individual manager [Man, 2004]. In the network environment, such decision making is carried out by different stakeholders - representatives of autonomous network partners - who often have different interests as well as different professional backgrounds and cultures. These and other factors make the collaborative decision making more complicated.

The collaborative decision making of the diverse network stakeholders takes the form of a *negotiation process*. One of the problems during this negotiation process is that stakeholders usually use natural language to present and communicate their statements. However, the stakeholders often have different views on control problems and solutions, as well as different interests, which, when communicated in natural language, may lead to incomplete and ambiguous statements [Malone, 1987].

We turn to the field of requirements engineering [Sommerville and Sawyer, 1997], which offers *conceptual modelling* as a common approach to address this gap. The notion of a conceptual model comes originally from the cognitive science, which describes it as a model consisting of a number of symbol structures and symbol structure manipulators which are supposed to correspond to the conceptualization of the world by human observer [Borgida et al., 1982]. Requirements engineering and computer science see a computer system as a model of a 'world' or a 'slice of reality' and conceptual models reflect the users' conceptualization of this world.

Conceptual models are based on the principle of *abstraction*, which is a fundamental tool in organizing information [Borgida et al., 1982]. Classification, aggregation and generalization are just a few aspects of abstraction useful to describe conceptual models. Classification refers to grouping entities that share common characteristics. Aggregation refers to treating a collection of concepts as a single concept. Generalization refers to extracting from one or more given classes the description of a more general class that captures the commonalities but suppresses some of the detailed differences in the description of the given classes.

Conceptual models are often supported by graphical interfaces, which can be displayed and discussed during negotiation meetings and brainstorming sessions with the stakeholders. The use of graphical conceptual models has proven to be useful in software engineering, which employs the models for representing software requirements. In particular, the utilization of graphical modelling techniques in software engineering improves the readability of complex formal specifications [Brathwaite, 1980]. In addition, graphical modelling techniques can be used to allow for greater user participation, which can lead to better acceptance of a new system and can therefore reduce the initial learning curve [Brathwaite, 1980]. Graphical conceptual modelling techniques are also used for business analysis [Tapscott et al., 2000, Weill and Vitale, 2001]. These features of conceptual modelling are very essential for our methodology, since we intend to develop a methodology used by *human* analysts.

Recently, some conceptual modelling techniques have been developed in the business science domain (see [Osterwalder and Pigneur, 2002], [Gordijn and Akkermans, 2003], for an overview [Pateli and Giaglis, 2004]). They have different purposes, for instance to assist in the discovery of new business models or to analyze their feasibility. An important requirement for conceptual models is that they represent a terminology common to the intended users in order to achieve a *shared understanding* among them [Borst et al., 1997]. This feature makes conceptual models attractive for the design of inter-organizational controls, since shared understanding is essential for decision making in a multi-enterprise environment.

The use of conceptual modelling is not new to the control field. As already mentioned, internal control practitioners utilize thorough conceptualization of business processes and information systems to get an understanding of the risks and necessary controls in internal business processes (see e.g. [Romney and Steinbart, 2006]). The components of procedural controls, such as evidence documents, data repositories, and the relationships between them can be modelled using *process modelling techniques*, such as flow charts and Data Flow Diagrams (DFDs). So, the conceptual modelling approach can also be useful for the design of inter-organizational controls.

2.2.1 Value models

The design of inter-organizational controls can be very complex, among others because different perspectives of the multiple stakeholders are involved. In requirements engineering, a *multi-viewpoint approach* is suggested as a mechanism to deal with this multi-perspective problem. By introducing several viewpoints on one system, complicated requirement issues can be broken down into self-contained perspectives, which can then be addressed and decided upon independent of each other [Finkelstein et al., 1992], [Finkelstein et al., 1994].

Requirements engineering distinguishes a *process viewpoint* and an *information system viewpoint*. The process viewpoint describes business processes and the information system viewpoint describes information systems that enable and support the business processes. For example, [Wieringa et al., 2003] suggest a framework for integrated business process and application architecture design. The tools for conceptualizing the process and information systems viewpoints include UML activity and sequence diagrams [Fowler and Scott, 2000], flow charts, and data flow diagrams [Hollander et al., 1999, Romney and Steinbart, 2006].

More recently, many researchers have emphasized the necessity of taking into account business requirements for system developing. As a result, several viewpoints for so-called early stages of requirements engineering were developed, one such viewpoint being a *value viewpoint* [Gordijn and Akkermans, 2003]. The value viewpoint focuses on the way economic value is created, exchanged and consumed in a multi-actor network. The models considered in the value viewpoint are often referred to as *business models* [Morris et al., 2005], [Osterwalder and Pigneur, 2002].

Another approach for early requirements engineering is *goal-oriented requirement engineering*, which expresses the relationship between systems and their environments in terms of goal-based relationships [Yu and Mylopoulos, 1998]. In this work we focus on the value viewpoint, since it is specifically focused on economic requirements.

Conceptualization of the value viewpoint is a rather new field of research. The conceptual modelling technique for value viewpoint, considered in this thesis, is e^3*value* suggested by [Gordijn and Akkermans, 2003]. An e^3*value* model represents a network organization as a set of *actors* who exchange *objects of economic value* with each other. An example of e^3*value* model is shown in Figure 2.2. The network presented consists of a buyer, a seller and a tax office, each exchanging the objects of economic value: $Payment$, $Goods$, VAT and $Legal\ Compliance$. A buyer buys goods from a seller and pays in return; the seller pays a value-added tax (VAT) to the tax office, as a result of which he obtains a legal compliance, meaning that he is not charged with a fine.

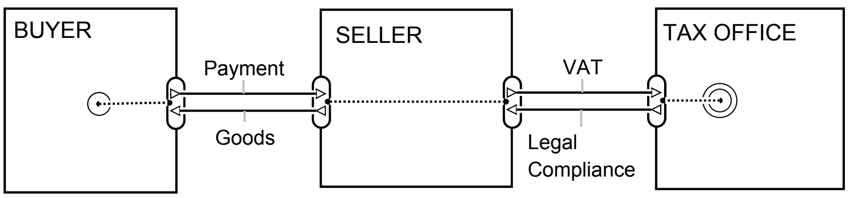


Figure 2.2: Example of an e^3*value* model of a purchase including a tax payment

The e^3*value* model abstracts from the time-dependence between operational activities, as often considered by the business process models, and focuses purely on transfers of objects of economic value. In this respect, the process viewpoint focuses on a way of putting the value viewpoint in operation in terms of business processes. For example, the value transfer of the object $Goods$ and $Payment$ between a buyer and a seller may be put into operation in different ways, e.g. a simple simultaneous exchange in a supermarket, a buyer pays first, a seller delivers first, or a complex Letter of Credit procedure, involving intermediaries [Kartseva et al., 2004a]. In fact, these decisions may reflect control considerations.

There are other conceptual modelling techniques for value modelling besides e^3*value* . We will briefly review them in Chapter 3.

2.2.2 Value viewpoint on controls

The internal control theory, the development of which dates back to the early days of information systems engineering, only considers process and information systems viewpoint which is a subject in the field of IT Audit, also called EDP Audit [ISACF, 2005]. A value viewpoint on controls does not exist in the internal control theory. We argue that the consideration of a value viewpoint when designing control mechanisms for networks is an important and useful. There are several reasons why this is so.

Controls as commercial services with added value. Understanding the impact of controls on business models requires the development of a value viewpoint. As we have observed in a number of case studies [Kartseva et al., 2004a, Kartseva et al., 2005a, Kartseva et al., 2005b, Kartseva et al., 2006b, Kartseva et al., 2006c], controls impact on business models in many different ways. Firstly, controls are essentially services with their own price tag (e.g. TRUSTe's web site seal[10]). They require financial resources, which affects the sustainability of a network. A value viewpoint can help in making different design choices when introducing controls, e.g. who is going to pay for a control service? who is going to execute it? and how will it affect the financial performance of network participants? In addition, a cost-benefit analysis of controls, which is an important part of a decision on a control structure, can be supported by value models. Furthermore, a need for control often leads to new business opportunities for control-related services. For example, in our analysis of health care quality controls in the Netherlands we developed a new business model of community-based quality control by means of an interactive web-site [Droes et al., 2005], [Kartseva et al., 2006c]. These business aspects of controls are not the subject of a process or information system viewpoint, and can be better highlighted in a value viewpoint.

Value aspects of control instruments. Many components of controls have an inherent value aspect. Examples we refer to as *control instruments* include evidence documents, rights and incentives.

The value properties of *incentives* (see section 2.1.2) can be well represented by value models. A reward or penalty can be modelled as a value object with an associated economic value. For example, an actor paying a monetary penalty can be modelled with an outgoing value object. The goal of an incentive scheme is to change the behaviour of an actor, e.g. by introducing a monetary incentive which decreases in the case of opportunistic behaviour. So, the value perspective enables us to see how an incentive impacts on the profitability of actors. This is not possible to model explicitly in process models or information systems models, since they do not have a notion of value in the first place.

Another example of a control instrument with a value aspect is a so-called *negotiable document*. Documents, used for control reasons, often have an economic value and can be traded. An example of such a negotiable document is the Bill of Lading, which is a document that is issued by a carrier to prove that he received goods from a seller, and that must be presented by the buyer to the carrier to claim the goods from the carrier [Kartseva et al., 2004a]. Bills of Lading in the oil industry are often sold many times when the oil is transported over sea. In these transactions the transfer of the Bill of Lading means a transfer of value (payment in return for bill of lading) as well as a transfer of an evidence control document (proof of ownership). Understanding

[10]TRUSTe is an independent, nonprofit organisation that certifies and monitors web site privacy and email policies, monitor practices, and resolve thousands of consumer privacy problems every year.

this complex relationship requires the use of a value perspective on controls. The negotiable documents are explained in more detail in Chapter 12.

The value viewpoint is an abstract rationale for procedural aspects of controls. The value viewpoint has two purposes for procedural controls: (1) to represent a high-level economic motivation of controls and (2) to provide abstraction of procedural details of controls.

The value perspective represents a high-level economic motivation of controls. Fraud and embezzlement often have an economic motivation: people steal to increase their wealth. Before designing procedural controls, the control problem should be understood from the economic perspective. Value models can provide a high-level specification of opportunistic behaviour and explain the economic motivation behind it.

Representing a motivation for controls from the value perspective makes the primary reason for controls more clear. The primary reason for introducing controls is always related to value, and not to processes. For example, consider the number of export documents within the EU. During our interviews with domain experts within the Beer Export case study, it became apparent that no one knows why the documents are needed. We therefore argue that before introducing the procedural controls, such as the documents, one should identify why the control is needed in the first place. Only if the absence of the control leads to a loss of some value, should such control be introduced.

The second purpose of the value viewpoint for procedural controls is motivated by the complexity of the networks. The design of inter-organizational controls can be very complex, because, among other things, multiple stakeholders have different perspectives. As already discussed above, a way to deal with this multi-perspective problem is to use a multi-viewpoint approach. By introducing a value viewpoint, we provide a high-level model of controls, without focusing on procedural details yet. Then, the procedural details can be designed to operationalize the value perspective.

2.3 Control patterns

What has not been discussed yet is how the available knowledge about each inter-organizational control should be organized. By this we mean the selection of a framework for describing the *design knowledge* about controls. The design knowledge of inter-organizational controls includes guidelines on how a business model or process in a current network organization should be changed in order to ensure sufficient control. There are several requirements for describing this design knowledge.

Firstly, we need to describe not only a control mechanism, but also a related control problem. A control mechanism is useless without the knowledge of the related control problem, since not all control mechanisms can mitigate every control problem. For example, monitoring helps against the hidden action problem, but not against the hidden information problem.

Secondly, the description of control problems and mechanisms should be supported by a graphical conceptual model to allow for easy communication with stakeholders and improve shared understanding. Such graphical representation is essential since we aim to develop a design methodology. The advantages of graphical conceptual modelling were discussed in 2.2.

Furthermore, we wish to address both value- and process-level aspects of control problems and mechanisms. The value aspects are needed to understand the economic motivation behind the control mechanism and its impact on the business model of the network. The process perspective is needed to design procedural aspects of controls, as in the internal control theory.

Patterns

To structure knowledge about inter-organizational controls we suggest using *patterns* which form are a well-known approach to structure complex design knowledge. A pattern is a description of a general and accepted solution for some recurring problem. Patterns prove to be useful for many reasons. They describe solutions for 'real world' problems, capture domain expertise, document design decisions, reuse wisdom and experience of master practitioners, and form a shared vocabulary for problem-solving discussions.

The pattern approach was originally proposed in architecture by [Alexander, 1979]. Alexander describes a pattern as *"a problem which occurs over and over again in our environment and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice"*. Patterns have proven to be successful in software engineering [Gamma et al., 1995], [Coplien and Schmidt, 1995], [Manns, 2001], where people recognized that much of the code they wrote is reusable in other contexts. Recent research on patterns in the business domain explores the idea of reusing patterns to solve recurring business problems (see [Tapscott et al., 2000], [Weill and Vitale, 2001], [Fowler, 1997], [Martin et al., 1998], [Rolland et al., 2000]).

There are many definitions of patterns in literature. Although most definitions of them are for the software engineering domain [Gamma et al., 1995], [Fowler, 1997], [Coplien and Schmidt, 1995], there are some definitions for the business domain as well. For example, [Rolland et al., 2000] in their work on patterns of organizational change argue that patterns should focus on specific problems within the context of organizations. They define patterns in business domain as "*generic and abstract organizational design proposals*".

It is important to realize that knowledge captured by patterns is already *available knowledge* rather than knowledge to be invented by the patterns designer. Patterns make it possible to capture recurring business problems and solutions, so that these solutions can be subsequently reused, allowing other practitioners to benefit from the experience of the pattern designers [Seruca and Loucopoulos, 2003].

Control patterns

We propose a library of control patterns which structures existing knowledge about inter-organizational controls. The advantage of a pattern approach for structuring knowledge about inter-organizational controls is that the traditional structure of a pattern coincides with the structure of the internal control taxonomy (see section 2.1.1). Traditionally, a pattern has the following structure: *name*, *context*, *problem*, *solutions* [Gamma et al., 1995]. In the control domain, *control mechanisms* can be seen as solutions for *control problems*. We can therefore define a *control pattern* as a *description of generic and re-usable control mechanism for a recurring control problem.*

The primary intent of a control pattern library is to provide useful abstractions of existing solutions to recurring control problems, for reuse. From a theoretical perspective, control patterns are innovative in several ways. Firstly, control patterns provide a structured way of designing controls, using conceptual modelling techniques well-known from software engineering. Unlike in the internal control theory, we take a value perspective in addition to the usual process perspective to model business networks by the transfer of economic value between participants. This also allows the designer to evaluate the impact of a control pattern on the business model of the network organization. Secondly, the library of control patterns that we present here is specifically tailored for inter-organizational control problems which deal with the risks of conducting a transaction in a network setting between parties that may not trust each other. Thirdly, our contribution extends previous research on inter-organizational controls [Bons, 1997], on which some of the patterns are based, by providing specific guidelines for the design of controls to deal with a larger set of control problems.

2.4 Further steps

In this section, we summarize the literature review and identify further steps of our research. We aim to develop a conceptual modelling approach to designing inter-organizational controls. As already argued, this approach must have the following characteristics:

- Address the design of control from the economic value perspective;
- Address the design of controls from the network perspective;
- Provide conceptual modelling techniques to support a human business analyst in the design process. The identified conceptual modelling techniques include *ontologies* and *patterns*.

None of the theories and methodologies reviewed contain all these three features. The internal control theory has some elements of the design methodology on controls. For example, the segregation of duties and other control principles described in section 2.1.1, are prescription-type rules on the structure of controls. In addition, the internal control theory employs conceptual modelling techniques, such as flow charts, to describe and audit administrative processes in organizations.

Furthermore, the focus of the internal control theory is primarily on internal control measures rather than on structuring of inter-organizational processes. Internal control theory does take some inter-organizational aspects into account in a way that an organization is always a part of some external environment [Romney and Steinbart, 2006]. Nevertheless, its main focus is on relationships within an organization and it is not that straightforward whether the internal control principles can and should be applied for inter-organizational relationships and to what extend.

The agency theory provides a clear structure and description of typical inter-organizational control problems and mechanisms. It describes inter-organizational controls. We have discussed some of them, namely *screening, signalling, monitoring* and *incentives* (see Figure 2.1). Unlike the internal control theory, the agency theory does not provide any conceptual design methodology at all. It does not provide prescriptive rules for the design of inter-organizational control systems. The agency theory research is mostly concerned with the selection of appropriate controls and on the empirical examination of performance of controls, e.g. by addressing such issues as what defines a better fit of a control and in which circumstances a control should be applied [Eisenhardt, 1989]. In our methodology, the reasoning about finding the best controls is assumed to be done by the human user, while the methodology provides the concepts to describe and structure the controls in a precise and accurate way for the human designer.

By combining findings from the internal control theory and the principal-agent framework, [Bons, 1997] provides some prescriptive design rules for inter-organizational controls, as shown in section 2.1.3. Only *monitoring* controls from the agency theory, have been considered by Bons. In addition, Bons identifies *evidence-based controls*, which stem from the internal control theory. One of the main results of [Bons, 1997] is a Petri Net-based expert system, which allows the encoding of inter-organizational transactions in a formal way and auditing their trustworthiness by checking their compliance with certain heuristics. Unlike Bons, we do not intend to develop an expert system which uses automated reasoning to select the optimal controls for a particular network. Instead we wish to develop a methodology to assist humans in the design process of controls.

As a result, we have undertaken the following steps. We start with developing a conceptual modelling technique to design controls from the value perspective. This is done in Chapter 3. This technique is based on the e^3*value* ontology discussed in this chapter previously. In Chapter 4 and 5 we test the conceptual modelling technique in two case studies. Furthermore, we develop control patterns. For this, in Chapter 6 we make an overview of the controls mechanisms extracted in this chapter and, after synthesizing them, we suggest a set of non-overlapping and unique problem-solution pairs of controls, which form the control patterns. The control patterns are further tested in two case studies in Chapter 12 and 13.

2.5 Summary

In this chapter we have introduced the concept of inter-organizational controls and defined our requirements for the design approach for inter-organizational controls. We argue that the approach should have the following features:

- Address the design of control from the economic value perspective;
- Address the design of controls from the network perspective;

- Provide conceptual modelling techniques to support a human business analyst in the design process. The identified conceptual modelling techniques include *ontologies* and *patterns*.

Designing controls from the value perspective is a core contribution of this thesis. In short, there are three motives behind introducing a value perspective on controls:

1. Controls are commercial services with added value;
2. Many control instruments have inherent value aspects;
3. Value viewpoint is an abstract rationale for procedural aspects of controls.

Furthermore, we have suggested a need for a pattern approach to organize knowledge about inter-organizational control mechanisms. The patterns capture the knowledge about the most common inter-organizational controls.

In the literature review in this chapter, we have considered the internal control theory [Hollander et al., 1999], [Romney and Steinbart, 2006], the agency theory [Eisenhardt, 1989] and the work on Bons inter-organizational controls [Bons, 1997], [Bons et al., 1998]. Although these theories provide a starting point towards the design of inter-organizational controls, no single source satisfies our requirements for the value-based design methodology of inter-organizational controls.

In the following chapters we develop a methodology for the design of inter-organizational controls in network organizations. In Part I we extend the e^3*value* model with additional concepts to represent opportunistic behaviour and control mechanisms on a value level. In Part II we develop a library of control patterns.

Chapter 3

The e^3*control* methodology

In this chapter we extend the e^3*value* methodology [Gordijn and Akkermans, 2003] so as to be able to analyze control problems and design controls in network organizations. The goal of such a methodology is to provide the tools that will improve understanding between stakeholders involved in the design of a control mechanism. The e^3*value* methodology is a suitable starting point for designing inter-organizational controls, since it allows us to model controls from the economic value perspective. As was argued in Chapter 2, the value perspective on controls is useful for several reasons. For example, because controls are basically services with their own price tag, they require financial resources, and therefore have effects on the financial sustainability of a network organization. In addition, the initial goal of a control is to safeguard a value. Finally, estimating the impact of controls on business models and vise versa requires a more rigid focus on economic value, which cannot be achieved with conventional process modelling techniques.

3.1 The e^3*value* methodology

Before extending the e^3*value* methodology, we describe some e^3*value* concepts in more detail using the example of a computer manufacturer (e.g. Dell) and its business network. Figure 3.1 shows a customer who buys a personal computer (PC) from Dell and pays money in return. Dell then does two things. Firstly, in order to assemble the PC, it buys various PC components from manufacturers. Secondly, Dell has to deliver the PC to the customer and to do this may choose between the services of two logistic service providers: UPS and DHL.

3.1.1 Basic elements

This scenario described above can be conceptualized in e^3*value* using the following e^3*value* constructs:

Actor. An actor is perceived by its environment as an independent economic (and often legal) entity. An actor makes a profit or increases its utility. In a sustainable business model

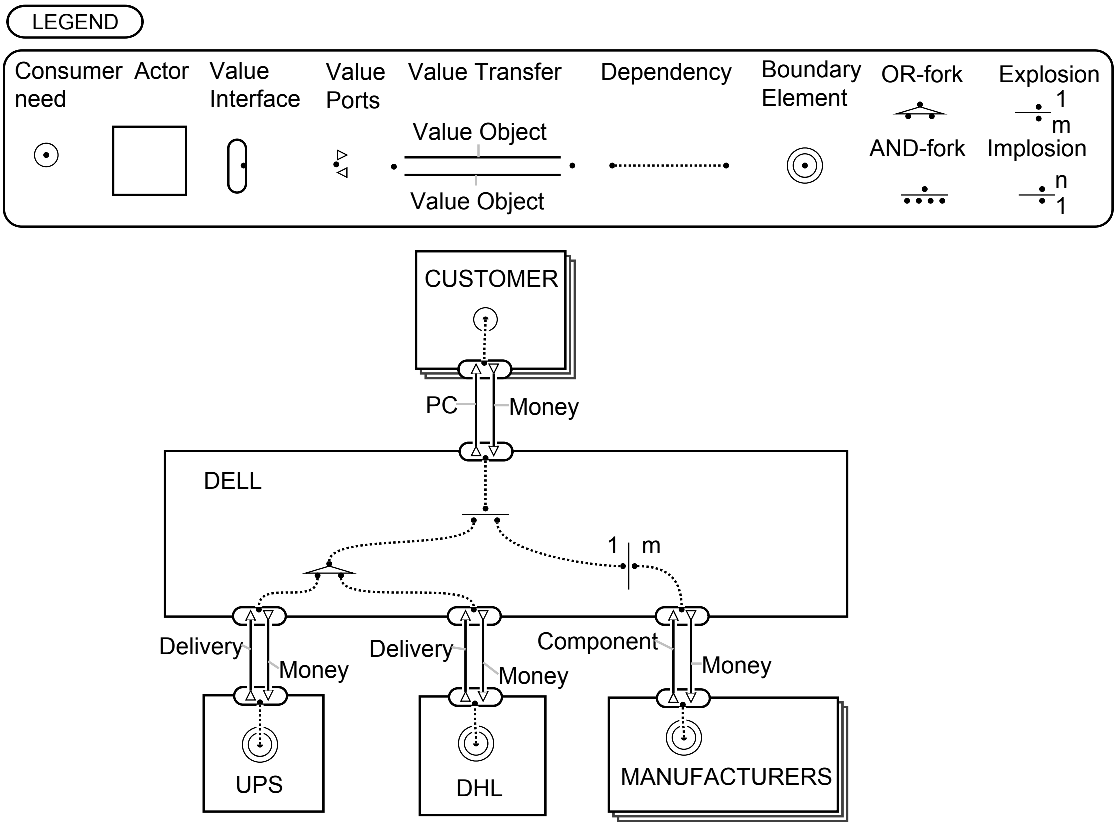


Figure 3.1: The e^3*value* model of a computer distributor Dell and its network

each actor should be capable of making profit. The example shows a number of actors: *Dell, UPS*, and *DHL*.

Market Segment. A market segment is a concept that breaks a market (consisting of actors) into segments that share common properties. Accordingly, the concept market segment shows a set of actors that assess value objects equally from an economic perspective for one or more of their value interfaces. In the example in Figure 3.1, the market segments are *Customer* and *Manufacturers*.

Value Object. Actors exchange value objects, which are services, products, money, or even consumer experiences. The important point here is that a value object is of value to one or more actors. In Figure 3.1 examples of value objects are *PC, Money, Components* and *Delivery*.

Value Port. An actor uses a value port to show to its environment that it wants to provide or request value objects. The concept of port provides an abstraction away from internal business processes, and thereby focusses only on how external actors and other components of the business model can be 'plugged in'.

Value Transfer. A value transfer is used to connect two value ports with each other. It represents one or more potential exchanges of value objects between value ports.

Value Transaction. A value transaction is the set of all value transfers in one value interface.

Value Interface. Actors have one or more value interfaces for grouping reciprocal, opposite-directed value ports. A value interface shows which value object an actor is willing to exchange *in return for* another value object via its ports. The exchange of value objects is atomic at the level of value interface.

The value interface concept allows us to represent an important assumption of *e^{3}value* , known as the **Principle of Reciprocity** which states that an actor is either willing to exchange a value object *in return for* some other value objects, or to make no exchange at all. The importance of the principle of reciprocity is that it represents the ideal behaviour of actors: an actor always offers something in return for obtaining a value object. Controls can be seen as measures that enforce such economic reciprocity.

3.1.2 Dependency path

With the use of the concepts introduced so far, we can explain who wants to exchange values with whom, but we cannot yet explain what happens in response to a particular end-consumer need. More specifically, we have to show how value transfers depend on each other. For this purpose *e^{3}value* uses dependency paths (based on [Buhr, 1998]) between value interfaces and inside an actor.

Consumer need. The start of a dependency path is a consumer need which is modelled graphically with a single-layered circle, also referred as a 'bull's eye'. In Figure 3.1, the consumer need indicates a need of the market segment *Customer* for persona computers *PC*.

Boundary element. A boundary element indicates where the dependency path ends which is modelled graphically with a double-layered circle, also referred as a 'bullet'. In Figure 3.1, there are three boundary elements: one at *UPS*, one at *DHL* and one at *Manufacturers*.

Dependency node. A dependency node is a consumer need, a value interface, an AND-fork, an OR-fork, an implosion, and explosion, or a boundary element.

Dependency segment. A dependency segment connects dependency nodes and it is represented by a link. The segment that connects two value interfaces indicates that if one interface exchanges its value objects, another interface also has to exchange its value objects. For example, the value interface of the transaction of *PC* and *Money* between *Dell* and *Customer* is connected via the interface to the value transfer of *Components* and *Money* with *Manufacturers*. This means that *Dell* cannot make money by selling a *PC* unless it first purchases the components.

AND-fork and AND-join. An AND-fork splits a dependency segment into two or more segments. An AND-fork is represented by a line, which has one incoming segment and two or more outgoing segments. In Figure 3.1, an AND-fork splits a segment coming from the interface of *Dell* with *Customer* into two segments: one leading to *Manufacturers* and one leading to *UPS* and *DHL*. This means that in order to deliver *PC* to *Customer*, *Dell* needs both exchanges with *Manufacturers and* with *UPS* or *DHL*. An AND-join is a 'reversed' AND-fork. It collects several segments into a single segment. It is represented by a line, similar to an AND-fork, but in this case has with several incoming segments and one outgoing segment.

OR-fork and OR-join. An OR-fork models a continuation of the dependency segment in a direction that may be chosen from a number of alternatives. An OR-fork is represented by a triangle, which connects one incoming segment and two or more outgoing segments. In Figure 3.1, an OR-fork splits a segment in two: one leading to *UPS* and the other one leading to *DHL*. This means that in order to receive an object *Delivery*, *Dell* can choose either *UPS or DHL*. The OR-join is a 'reversed' OR-fork. It merges two or more segments into one segment. As with an OR-fork, it is also represented by a triangle, but with several incoming segments and one outgoing segment.

Dependency path. A dependency path is a set of dependency nodes connected by segments that lead from a consumer need to a boundary element. In Figure 3.1, the path starts with a consumer need at *Customer* and ends after the value transfers with *DHL* or *UPS*, and with *Manufacturers* are completed.

Implosion and Explosion. If we return to the out-going port on the far-right of the AND-fork in Figure 3.1, we see that it is connected to an *explosion/implosion* element. Such an element contains exactly one in-coming port and one out-going port with different occurrences. It is called an *explosion element* if the occurrences on the out-going port are higher than on the in-coming one, otherwise it is an implosion element. In this example, the element shows that in order to deliver one *PC* to a customer, *Dell* must purchase m components at *Manufacturers*. So, the occurrences at the in-coming port, connected to the value object *PC*, are 1 and the occurrences at the outgoing port, connected to the value object *Components*, are m ($m > 1$). This means that this is an explosion element.

3.1.3 Valuation

The *e^{3}value* models allow us to estimate the profitability of each actor in a value model. To do this, we have to estimate the number of actual value transfers in a time period (e.g. a year) and the monetary value of the transferred value objects. For each actor, the results are summarized on a *net value sheet*, which shows an estimate of whether the business model is or could be profitable. We assume that a value web can only be sustainable in the long term, if each actor can create a positive net value flow.

Valuing value objects. To calculate the net value flow sheet we need to assign economic value to the objects obtained and delivered. If we know how many objects are transferred, and their economic value, we can construct a net value flow sheet. An example is given in Table 3.1.

How to value a value object? The *e^{3}value* methodology distinguishes two types of value objects: *monetary objects* (these are monetary objects such as banknotes, checks and alike) and *non-monetary objects* (e.g. goods or services).

The reason for distinguishing monetary and non-monetary objects is that the valuation of these objects differs, as *e^{3}value* argues. If money objects are transferred, it is possible to state *objectively* how much money is transferred. Perhaps the amount of money to be paid is on a suppliers price-list or the outcome of a negotiation process, but ultimately the amount of money transferred can be objectively observed by each participant in a value web; they do not disagree about this amount of money.

Each actor assigns their own *subjective* economic value to non-monetary value objects. For example, if we ask a number of people to value the same sports car in terms of monetary units (how much money each person is willing to pay for the sports car), each person will probably come up with a different value. The valuation of non-monetary objects also depends on context like place and time. For instance, many people assign a higher economic value to water if they are alone in the dessert, than if they are in an area where they can easily buy water. To sum up, people assign value to non-money value objects differently. The valuation of non-monetary objects allows us to take the concept of utility [Varian, 2006] into account. In modern economics, utility is a way of describing *consumer's preferences* which are subjective and can therefore be modelled in *e^{3}value* by using the non-monetary values. These aspects are useful in the valuation of penalties and incentives in Chapter 9.

Example. The software tool support for *e^{3}value* allows us to assign numbers to value objects and calculate the net value sheets for each actor in a value model. Table 3.1 shows such a net value sheet for Dell in the example given earlier. The first column lists all the actors that have value transactions with Dell. The second and the fourth columns list the formulas for calculating the value of incoming and outgoing value objects of Dell with each actor correspondingly. The

Table 3.1: The net value sheet for Dell in Figure 3.1

Actor	Dell			
	Value Object In	Valuation	Value Object Out	Valuation
with Customer	*Price PC X Volume*	$20,000 M	(PC)	
with Manufacturer	(Components)		*Price Component X Volume*	$10,000 M
with DHL and UPS	(Delivery)		*Price Delivery X Volume*	$1000 M
Total net value flow	$9000 M			

third and the fifth columns list estimations of these formulas for the incoming and outgoing value objects correspondingly.

We estimate the monetary value of each value object (non-monetary objects are left out). To calculate a net value for a specific actor we subtract all his out-going value objects from all his in-coming value objects and multiply the result by the number of times the value object is transferred. For example, the valuation of the object *Money* coming from *Customer* to *Dell* is estimated as the price of each computer sold *Price PC* times the amount of computers sold *Volume*. If Dell sells a computer to one customer for $1000 and if around 20 million PCs are sold annually [1], then the value object *Money* is estimated as $1000 X 20 M = $20,000 M. The valuation of object *Money* for the transfers between Dell and manufacturers/distributors (UPS and DHL) is carried out in the same way. If Dell buys components for $ 500 and pays $50 for the delivery for one computer, then for 20 million computers the components will cost $10,000 M and delivery $1000 M.

Finally, profit or loss is calculated as the difference between the total value of the incoming and the total value of the outgoing value objects. In our example, Dell's net value is ($20,000 M - $10,000 M - $1000 M) = $9000 M per year.

Note that in our estimations we make many simplifications. More fine tuning can be done to achieve more precision. For example, a distinction could be made between different market segments, different pricing models of components manufacturers (we only estimate total spending for the components), and different pricing models of logistic providers (we assume they are the same).

Sensitivity analysis

The *e*^{3}*value* methodology can be used to perform sensitivity analysis of different business models. For example, an analysis can be made of how certain changes in prices or market share influence the profitability of each actor.

Such a sensitivity analysis could be performed for controls to make *cost-benefit analysis*. The situation could be modelled with and without controls and the financial impact of both could be

[1]The data taken here are fictional

compared. To do this, it would also be necessary to estimate the probabilities of the occurrence of certain control problems. In *e^{3}value* these probabilities could be included as choices at the OR-fork. Although this is an interesting and important line of research, we will not focus on the cost-benefit analysis any further in this thesis and leave it for future research.

3.1.4 The *e^{3}value* ontology

The basis of the *e^{3}value* methodology is the *e^{3}value* ontology shown in Figure 3.2. It is represented as a UML class diagram, which models relationships between the *e^{3}value* concepts.

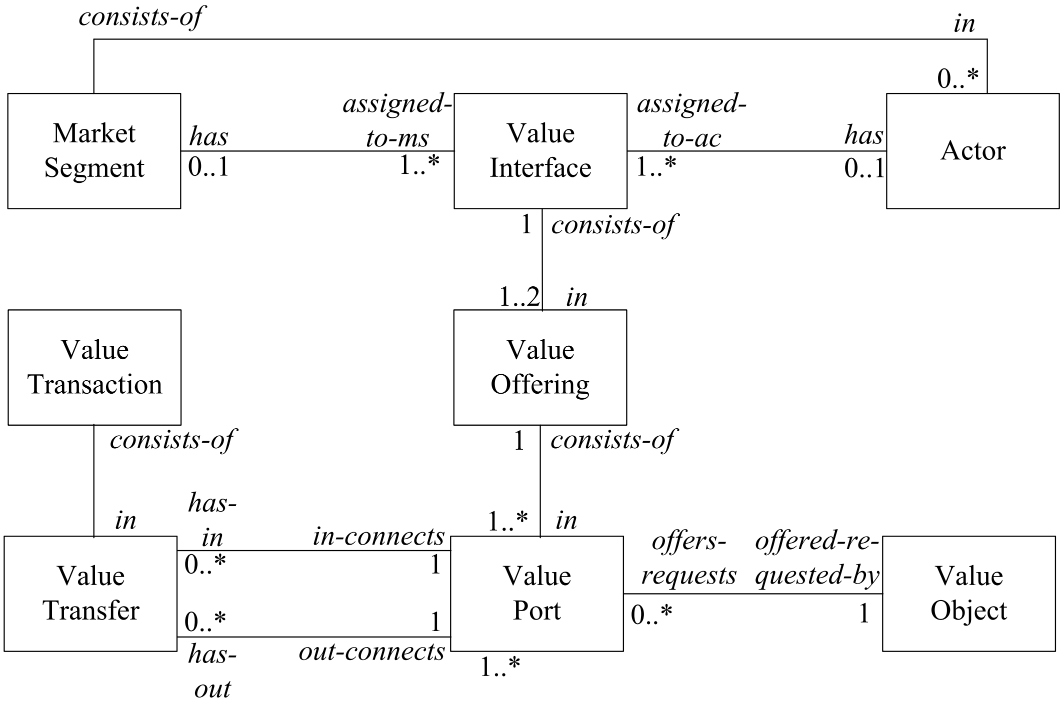


Figure 3.2: The *e^{3}value* ontology, source [Gordijn, 2002]

Figure 3.2 contains the *e^{3}value* concepts, already named: market segment, actor, value interface, value offering, value port, value transfer, value transaction, and value object. The links between the objects show how the elements are related with each other. The value interface is *assigned to* zero or one actor and it *consist of* one or two value offerings, which *contain* at least one value port. A value port *contains* zero or more in- and out-going value transfers, which *constitute* a value transaction. Finally, a value port can *offer* or *request* one value object.

3.1.5 Comparison with other approaches

Apart from *e^{3}value*, various conceptual modelling techniques have been developed to analyze and reason about the *economic aspects* of network organizations. Other alternative value-oriented ontologies are Business Modelling Ontology (BMO), developed by Pigneur and Osterwalder, expressing the business logic of firms [Osterwalder and Pigneur, 2002]; value webs of [Tapscott et al., 2000], and finally, Resource-Event-Agent ontology (REA), developed by Geerts and McArthy, which takes an accounting view on the economic relationship between various economic entities [McCarthy, 1982]. These value ontologies focus on the creation and transfer of economic value and are often called *business modelling* ontologies, as opposed to so-called *enterprise modelling* ontologies such as the Enterprise Ontology [Uschold et al., 1998] and the Business Process Handbook [Malone et al., 1999] that are more focused on organizational structure, activities and management.

We argue in favour of the *e^{3}value* ontology to represent a value viewpoint on controls. To this end, a number of design choices which appeal to us have been made in developing the *e^{3}value* ontology. This is a lightweight, rigorous ontology with a graphical representation, formally specified in such a way that software support tools can be developed for it. It focuses on a network organization, which is also a focus of our study. Usability of *e^{3}value* has been successfully validated in a variety of industries including entertainment, news, internet service provisioning, banking, trade documents and energy supply [Gordijn and Akkermans, 2001, Gordijn and Akkermans, 2003, Gordijn, 2002]. In these and other case studies, *e^{3}value* has proved to be a useful support for decision making in groups [Kartseva et al., 2004b]. We now go on to discuss the arguments in favour of *e^{3}value* in more detail by comparing it to other existing ontologies.

1. Firstly, *e^{3}value* is a *lightweight* ontology, meaning that the ontology contains only a small number of concepts and relationships. As a result, it can easily be understood and explained. Creating a business model using the ontology can also be done within a reasonably short timeframe, which is a requirement given time-to-market considerations. Many other ontologies are too extensive, and therefore less useful. For example, the Business Process Handbook project [Malone et al., 1999], an ontological business process modelling approach, developed for designing organizations, has a conceptual model with about 3400 different activities with 20 levels of specialization and 10 levels of decomposition. Such a complex conceptual model is too complicated for exploring network organizations, where the modelling object is not just a single organization, but rather a large group of organizations. BMO has more concepts than *e^{3}value* . REA also has a small number of concepts. Value webs of Tapscott are lightweight, but less formal (see below).

2. Secondly, to represent instantiations of the ontology (i.e., particular business models), a *graphical syntax* has been developed using a software tool support (available from http://www.e3value.com). As was argued in Chapter 2, a graphical representation is an important feature to enhance communication with business stakeholders. BMO and REA do not have a fully developed graphical representation. Value webs of Tapscott are graphically represented, but no systematic method is given how to make these representations. They do not specifically focus on the economic value perspective and contain some elements of process modelling.

3. Thirdly, the focus of *e^{3}value* is on the creation, distribution, and consumption of economic value in a network of enterprizes and consumers. The terminology of *e^{3}value* comes from research on strategic perspective of network organizations [Tapscott et al., 2000],

[Porter, 1980], marketing [Kotler, 1988] and axiology [Holbrook, 1999]. The focus on economic aspects of a network and the relevant terminology is essential for our research. Although BMO also views the networks of suppliers and customers, it limits the notion of business model to a single firm. Tapscott's value webs do not specifically focus on the economic value perspective, but contain some elements of e.g. process models. REA considers the external activities of a firm, although its original purpose was to develop internal Accounting Information Systems. The disadvantage of REA for our task is its limitation within the accounting definition of value, e.g. unlike *e^{3}value* it does not consider 'experience' to be a value object. As the name suggests, the focus of REA is on resources and events, which are more oriented towards a process viewpoint. In addition, REA is a structural, and not a behavioral model (see below).

4. Fourthly, the *e^{3}value* ontology has been formally described to enable the development of software tools that support automated analysis and the evaluation of designed business models. The *e^{3}value* model enables computer-supported reasoning about financial sustainability by generating net value flows in the form of a spreadsheet analysis for each actor involved. The *e^{3}value* ontology is available as a UML class diagram, RDF/Schema implementation, Prolog rule base, and Java-programmed design workbench (see http://www.e3value.com/). Tapscott's value webs are very informal.

3.2 The *e^{3}control* design framework

In this section, we introduce some steps which are usually needed to design controls. From the network perspective, we refer to these steps as the *e^{3}control* design framework. In addition, we introduce some terminology used throughout the thesis.

Based on the internal control theory, as described in section 2.1.1, the process of design of controls consists of three steps. In the first step an analyst identifies a transaction cycle, which in fact represents a *normative model* on how a process should occur so that the risks of fraud and errors are minimized. In the second step, an analyst compares real processes with the normative one and identifies potential or present *control problems*. In the third step, the analyst designs *control mechanisms*.

In our opinion, a more precise terminology to describe these three steps is provided by deontic logic [Wieringa and Meyer, 1993], [Jones and Porn, 1985] in which two states of a network can be distinguished: *ideal* and *sub-ideal*. Applied to the domain of inter-organizational controls, a network in an ideal state is one in which all actors behave as they are expected to (e.g. according to contractual agreements). Such behaviour of the actors is known as *ideal behaviour* and the ideal state of a network is referred to as *ideal situation*. In a sub-ideal state, actors in the network do not behave ideally and this is called *sub-ideal behaviour*. According to the internal control theory, sub-ideal behaviour can be intentional (e.g. fraud) or unintentional (e.g. errors). The sub-ideal state of a network is referred to as *sub-ideal situation*. As a result, we suggest the following steps to design inter-organizational controls:

1. *Ideal situation*: Designing the ideal situation.

2. *Sub-ideal situation*: Analysis of existing or possible sub-ideal situations.

3. *Control mechanisms*: Designing control mechanisms to reduce damage caused by control problems.

These steps constitute the e^3*control* design framework. In the first step, we model an ideal situation in which we describe how we want the actors in a network to behave and we assume that no fraud or errors occur. This is similar to normative models of transaction cycles in the internal control theory. In addition, both a value view and process view on the ideal situation can be designed.

In the second step we describe situations, in which actors in a network behave sub-ideally, e.g. commit fraud or make unintentional errors which lead to loss of value of their counter parties. This is similar to control problem analysis in the internal control theory, however, in our case both the value view and process view of the ideal situation can be designed.

Finally, in the last step we address sub-ideal behaviour by adding control mechanisms. This model with controls is an 'improved' ideal model. It contains controls against a problem described in the second step. Again, we can model these situations from both the value and the process view.

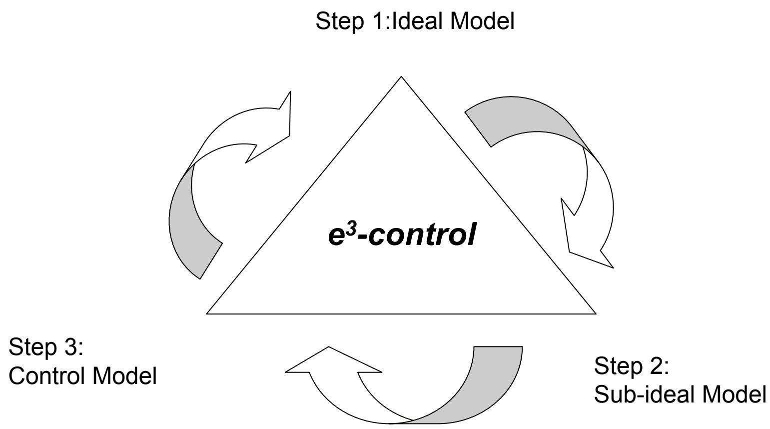


Figure 3.3: The e^3*control* cycle

A network usually contains several control problems. Therefore, the steps can be repeated in cycles to address each control problem. After a previous problem is addressed, the 'improved' ideal model can be taken as 'ideal' model in the next cycle, where the next problem is addressed. This creates a cyclic approach, shown graphically in Figure 3.3 and demonstrated in case studies in Chapters 4, 5, 12 and 13.

3.2.1 Subjective view on sub-ideal behaviour

An important issue here is who determines what behaviour is sub-ideal and what is not. We incorporate a **subjective view** on sub-ideal behaviour which assumes that such behaviour is defined subjectively by one actor or a group of actors. For example, from a customer's point of view an untimely delivery by a seller is sub-ideal. However, the seller might not share this view if we assume that he is a rational actor, interested only in the maximization of his own wealth.

However, actors rarely make decisions based only on rational concerns. Very often they define something to be sub-ideal based on existing *legal* or *social norms*. In the previous example, such a legal norm can be a contract between a customer and Dell that proclaims that late deliveries are sub-ideal. In this case, if the seller agrees with the contract (e.g. by signing it) he also considers the late delivery to be sub-ideal. The conventional use of *e^{3}control* is where various parties in a contracting situation jointly agree on what they consider the relevant control problems to be, hence the sub-ideal model normally represents the agreed control concerns of all parties involved. If no such control concerns can be agreed on, it means that a situation exists in which parties cannot agree on a contract.

Another example is a regulation to stimulate the production of renewable energy, as discussed in Chapter 12. In the past, the emission of carbon-dioxides was considered sub-ideal only by a limited group of environmentalists. The reduction of non-renewable energy production was not generally accepted by all groups of society, since there were (and still are) parties, who resisted renewable energy production, e.g. oil companies because it would reduce oil-based energy production. However, after the Kyoto protocol, the opinion of the environmentalist groups became the general policy of several governments. The Kyoto protocol defines a legal norm, accepted by the governments of countries who signed it, and as a result, the emission of carbon dioxide is seen as sub-ideal by the governments as well.

3.2.2 A need to extend *e^{3}value*

To address the three steps of the framework from the value viewpoint, we need to be able to model both ideal and sub-ideal situations in value models. The *e^{3}value* methodology presupposes an ideal world, in which parties in the value web do not commit a fraud or behave opportunistically with respect to each other. A value model expressed in *e^{3}value* can be seen as a model of a contract in which parties have obligations to exchange value objects and act in accordance with a contract. Therefore, *e^{3}value* can only model ideal situations.

The *e^{3}value* methodology is not enough to model sub-ideal situations. Take for example the *e^{3}value* model in Figure 3.4, where we show value transfers between a seller, a buyer and a tax office. The model shows that the buyer pays the seller for the goods and that the seller pays the *VAT* tax to the tax office in return for getting a compliance with the tax law. This is modelled with the value objects $Goods$, $Payment$, VAT and $Legal\ Compliance$.

One sub-ideal situation of this model is when the seller does not pay taxes when he should. In a value model, this corresponds to a transfer of a single value object *Legal Compliance* from the Seller to Tax Office (because the value object *VAT* is not transferred). But such a single value transfer is not allowed in *e^{3}value* because of the Principle of Reciprocity, which says that actors only exchange value objects in return for one or more other value object, or they do not exchange at all (see section 3.1). In *e^{3}value* terms, a transfer of a single value object is an invalid construct and the attempt to model it should result in an error message by an *e^{3}value* model checker.

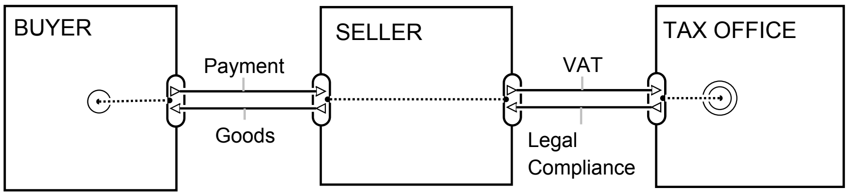


Figure 3.4: Ideal value transfers and value objects in e^3*control*

Since e^3*value* only models ideal behaviour, we need to extend it with concepts to represent sub-ideal behaviour. We call this extension ***e*^{3}*control***. An e^3*control* model enables representation of control problems as **sub-ideal value models**.

3.3 Constructs of the e^3*control* ontology

We extend the e^3*value* otology to the e^3*control* ontology in order to be able to model sub-ideal situations when the Principle of Reciprocity does not hold. The constructs of the e^3*control* ontology enable the modelling of sub-ideal situations. In particular, e^3*control* allows us to specify value transfers affected by a violation as well as the actors responsible for the violation.

Firstly, we represent a violation of the principle of reciprocity by introducing **sub-ideal value objects** and **sub-ideal value transfers**. In Figure 3.5 we model several sub-ideal situations that can happen when one of the actors violates the exchange obligations stated in the ideal model of transactions between a seller, a buyer and a tax office in Figure 3.4.

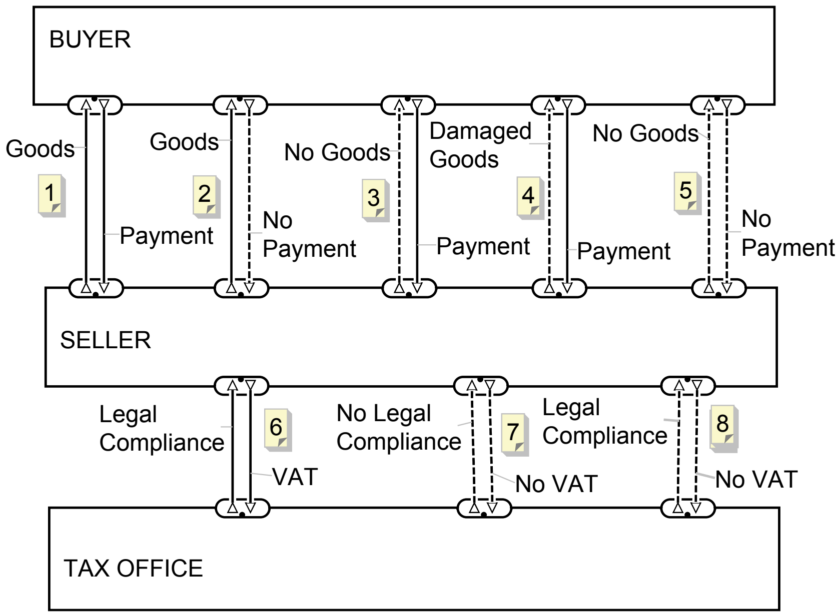


Figure 3.5: Sub-ideal value transfers and value objects in e^3*control*

The value transactions 1 and 6 in Figure 3.5 correspond to the ideal situation modelled in Figure 3.4. As in Figure 3.4, the actors exchange so-called **ideal value objects** $Goods$, $Payment$, VAT and $Legal\ Compliance$.

Other value transactions correspond to sub-ideal situations in which at least one actor does not behave as prescribed by the ideal situation. For example, value transaction 3 models that the goods have been paid for but not delivered. Failure to deliver the goods is modelled with the object $No\ Goods$, which is a **sub-ideal value object**. Its value transfer is marked with a dashed line. Other sub-ideal situations are modelled in a similar way: the sub-ideal value object $No\ Payment$ represents non-payment of goods, the object $Damaged\ Goods$ represents a delivery of damaged goods, the object $No\ VAT$ represents non-payment of VAT, and the object $No\ Legal\ Compliance$ represents a denial of legal compliance. In the value transaction 8, legal compliance is exchanged for tax payment while taxes are not paid. So, the $Legal\ Compliance$ object is sub-ideal, because the legal compliance should only be given if the taxes are paid.

We now go on to define the concepts used in 3.5 more precisely. These are sub-ideal value objects, sub-ideal value transfers, sub-ideal value ports, liability tokens, sub-ideal dependence segments and paths, and sub-ideal actors. The concepts and relationships between them are summarized in the UML class diagram in Figure 3.6.

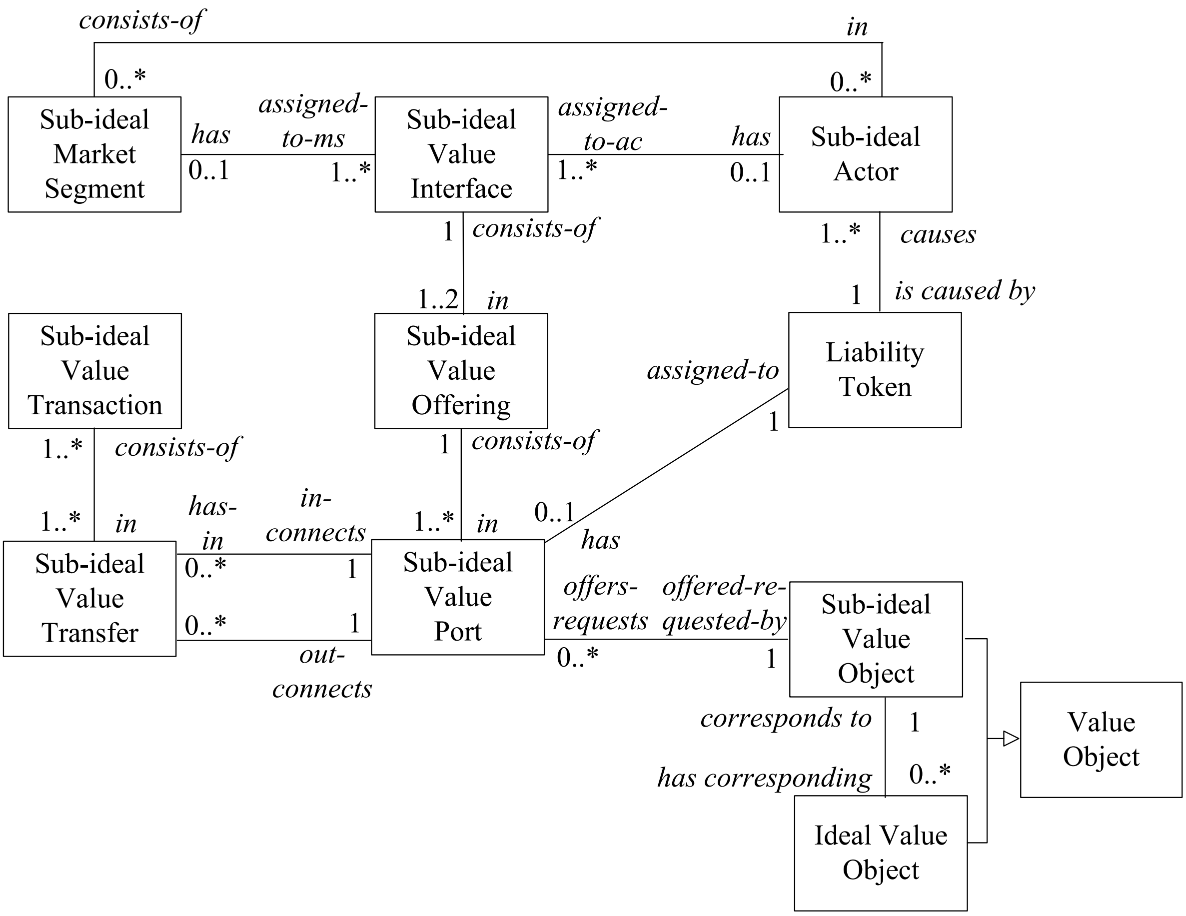


Figure 3.6: The *e3control* ontology

3.3.1 Sub-ideal value objects

A **sub-ideal value object** models a value object affected by sub-ideal behaviour. It usually has a name that is related to the name of its corresponding ideal value object. For example, if we name the ideal value object ‘Goods’, a sub-ideal value object can be named ‘Damaged Goods’. Because multiple control problems are possible, each ideal value object corresponds to multiple sub-ideal value objects. In Figure 3.5 the objects $No\,Goods$, $No\,Payment$, $No\,VAT$, $Damaged\,Goods$ and $No\,Legal\,Compliance$, and $Legal\,Compliance$ in value transfer 8 are sub-ideal.

3.3.2 Sub-ideal value transfers

A **sub-ideal value transfer** transfers a sub-ideal value object. It is visualized with a dashed line. In Figure 3.5 all the value transfers with sub-ideal value objects $No\,Goods$, $No\,Payment$, $No\,VAT$, $Damaged\,Goods$ and $No\,Legal\,Compliance$, and $Legal\,Compliance$ in transfer 8 are represented with a dashed line.

3.3.3 Empty and incorrect sub-ideal value objects

We can distinguish two types of sub-ideal value transfers: transfers of **empty value objects** and transfers of **incorrect value objects**. Empty value objects correspond to a situation in which *no* value is delivered to a receiving actor. The examples in Figure 3.5 are $No\,Goods$, when ordered goods are not delivered, $No\,Payment$, when no payment is made, and $No\,VAT$, when no taxes are paid. An incorrect value object is a value object *different* from the one transferred in a corresponding ideal situation. The examples in Figure 3.5 are $Damaged\,Goods$, when goods are delivered in a damaged state. A good with other characteristics than ordered is also an example of an incorrect value object.

3.3.4 Liability tokens

In Figure 3.5 we modelled six pairs of value transfers in which the Principle of Reciprocity does not hold. However, from this model it is not clear which actor is responsible for the violation, i.e. which actor behaves sub-ideally. For example, it is not clear if the seller is responsible for damaging the goods or the buyer.

To address this issue, we introduce a **liability token**, which is assigned to one of the two value ports of a sub-ideal value transfer and is assigned to an actor responsible for this sub-ideality. A liability token is visualized with a letter **L**, assigned to a value port that contains the sub-ideal value transfer. One actor may have multiple liability tokens, corresponding to different kinds of sub-ideal behaviour.

In Figure 3.7 we see liability tokens placed at various sub-ideal value transfers. In value transactions 2 and 5 the buyer is responsible for the sub-ideal value transfer $No\,Payment$. In value transactions 3 and 5 the seller is responsible for the sub-ideal value transfer $No\,Goods$. In value transaction 4 the seller is responsible for the sub-ideal value transfer *Damaged Goods* and so on.

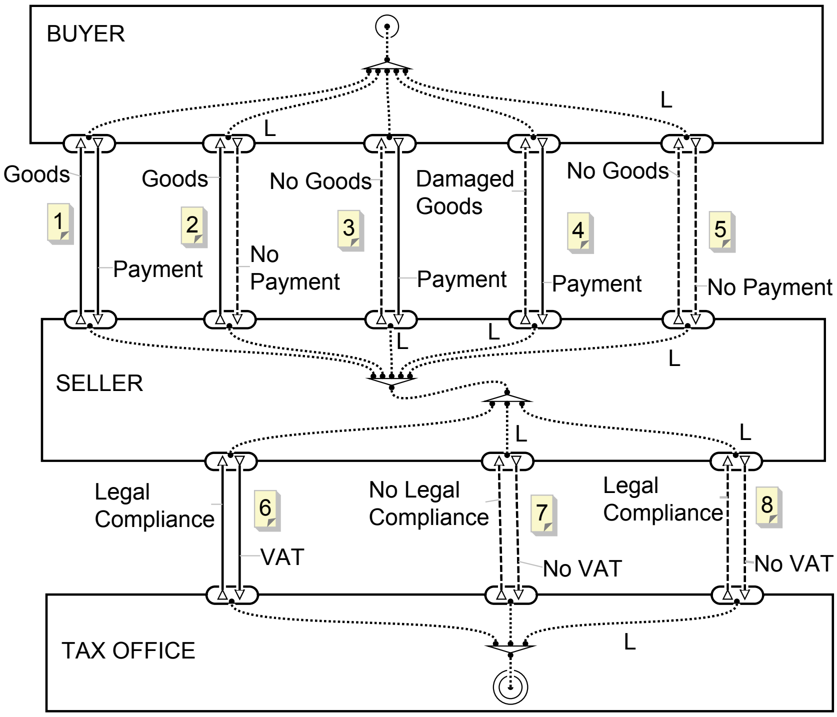


Figure 3.7: Liability tokens and sub-ideal dependency paths in e^3*control*

Note that e^3*control* does not give advice on how to identify a sub-ideally behaving actor. This is very domain- and case-specific and has to be determined by a modelled. e^3*control* only provides concepts to describe the sub-ideal behaviour.

3.3.5 Sub-ideal value port

Every value transfer, ideal or sub-ideal, has at least two ports: ingoing and outgoing. A value port is sub-ideal if it is connected to a sub-ideal transfer *and* has a liability token. Liability tokens are assigned to ports that contain sub-ideal value transfers, as explained above. So, all value ports which are connected to dashed lines, and that have liability tokens are sub-ideal. For example, in Figure 3.7 value transaction 4 models that a damaged good is the responsibility of the seller. The liability token can also occur on an incoming value port. For example, when damaged goods are the responsibility of the buyer, the liability token in the value transaction 4 should be modelled at the buyer and not at the seller.

3.3.6 Sub-ideal value interface

A value interface is sub-ideal if it contains at least one sub-ideal value port. In Figure 3.5 all value interfaces, containing value ports with liability tokens are sub-ideal.

3.3.7 Sub-ideal dependency segments

A **sub-ideal dependency segment** is a segment connected to at least one sub-ideal value transfer. Graphically, the segment is marked with a liability token at the specific port. In Figure 3.7 every segment that is connected to a value transfer marked with a dashed line is sub-ideal. These are segments connected to value transactions 2, 3, 4, 5, 7, and 8.

3.3.8 Sub-ideal dependency paths

We define an **ideal dependency path** as one that only contains ideal segments and has the same structure of segments and AND- and OR-forks as in the ideal value model. An ideal path models an ideal situation. Consequently, a **sub-ideal dependency path** is a path that contains at least one sub-ideal segment. In other words, it goes through at least one sub-ideal value transfer. Therefore, a sub-ideal path represents a sub-ideal situation.

We see several paths in Figure 3.7. Each one starts with a consumer need at the buyer and follows with an OR-fork. The OR-fork splits the initial paths into five sub-paths. Then at the seller we have a second OR-fork which leads to the three sub-paths. So, a total of fifteen paths are modelled in Figure 3.7. Of these fifteen paths only the path that goes through transfers 1 and 6 is ideal. All other paths are sub-ideal since they go through at least one sub-ideal value transfer. For example, the path that goes through value transfer 2 and 6 models a situation when the seller does not get payment from the buyer, but that he does pay the taxes. The path that goes through value transfers 2 and 7 models that the seller neither gets a payment nor pays the taxes.

3.3.9 Sub-ideal actors

Actors that have liability tokens behave sub-ideally. They can be called sub-ideally behaving actors, or just **sub-ideal actors**. In Figure 3.7 every actor has a liability token. Note that the token is assigned to a value port which means that only one actor is responsible for the sub-ideality of a transfer connected to a port with a liability token.

3.4 Modelling rights and evidence

In the remaining sections of this chapter we will address the issue of modelling rights and evidence documents in e^3*value* models. *Evidence documents* play an important role in controls and are often associated with economic value. Furthermore, evidence documents often represent *rights*, for example, the stocks, which are essentially evidence documents, represent rights for company's profit paid in a form of dividend. In addition, stocks have a value, defined by the market.

Taken that e^3*value* differs greatly from process modelling [Gordijn, 2002], it is sometimes difficult to decide whether the transfer of an evidence document, the transfer of a right or a physical transfer of some object are really *value* transfers. For instance, is an airplane ticket a value object, given that it cannot be sold or refunded?

In order to answer this question, we extend the e^3*value* model presented in Figure 3.2 in two ways. Firstly, we suggest that there is a need to distinguish between *possession* and *ownership rights*. Secondly, to relate value transfers to transfer of possession and rights, we we examine the operational details of a value transaction, which is the concept of *operational activity*. As a result, we bridge procedural aspects of controls of the internal control theory (see section 2.1.1) with the value aspects of e^3*value* . In addition, we also describe how the concept of evidence document fits into this relation.

3.5 An extended value object ontology

In Figure 3.8 we propose a model, which relates the e^3*value* concepts, such as 'value object' and 'value transfer', to process-level concepts, such as 'operational activity', 'right', 'custody' and 'evidence documents'. This model is based on findings in [Kartseva et al., 2006a]. Other authors suggest similar models [Andersson et al., 2006, Weigand et al., 2006]. In our model we include the concepts that are relevant for our goal to link the concept of evidence documents and rights with the concepts of value object and value transfer. New elements added to the initial e^3*value* ontology in Figure 3.2 are shown in gray. The new elements are *Resource*, *Right*, *Evidence Document*, *Custody*, *Activity*, *Transfer Activity* and *Conversion Activity*.

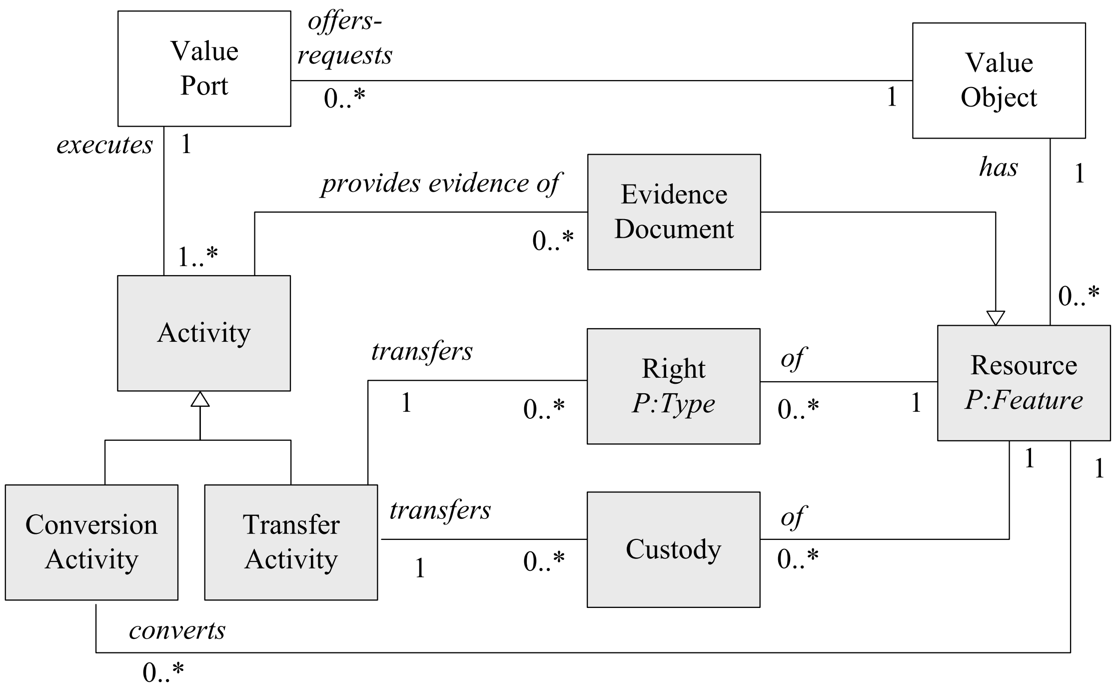


Figure 3.8: Extended value transfer ontology

We will now describe the concepts of the model in Figure 3.8 and then go on to discuss how rights and evidence documents can be modelled in e^3*value* .

3.5.1 Resources

Figure 3.8 shows that every *Value Object* is associated with one or more *Resources*. The concept of *Resource* is different than the concept of *Value Object* which refers to an abstraction of something of value. A *Resource*, on the other hand, is not necessarily of value to anyone. For instance, a *Resource* can play a role in a process model, but may not be of relevance to a value model.

A value object can require several resources. For example, a value object *Delivery Service* consists of the resources *Car* and *Driver*. A hair dresser service requires resources such as *Scissors, Hairdryer, Hairdresser, Electricity, Shampoo* and *Water*, to name just a few. Unlike a value object, a resource is not always of value to someone. For example, scissors alone are of no value to someone who wants to have a haircut. This person values the whole bundle of resources which are represented by a value object *Hairdresser Service*.

It is possible to identify some general categories of resources such as physical goods, information, and money. [Baida, 2006] also identifies human resources, capability resources, and experience resources[2]. Human resources refer to the human effort required for a value transfer to take place. Capability resources refer to an ability to use some service, e.g. insurance, which can be used or not. Experience resources represent an experience offered by a value transfer, e.g. an experience of a good service.

A resource may have properties and associations with other objects, e.g. the colour of a shirt. Such properties and associations are modelled by means of a property *Feature* of the *Resource*, see Figure 3.8.

3.5.2 Rights

According to the property rights theory [Alchian and Demsetz, 1972], [Coase, 1960], resources are associated with *property rights*. A property right is the exclusive authority to determine how a resource is used, whether that resource is owned by a government or by individuals [Alchian and Demsetz, 1972]. [Coase, 1960] suggests that each asset (or, as we call it - a resource) refers to a *vector of rights*. Conventionally, property rights include (1) exclusive rights to the use of a resource, (2) exclusive rights to the services of a resource, and (3) rights to exchange the resource at mutually agreeable terms [Alchian and Demsetz, 1972]. Other types of rights include copyrights.

So, in Figure 3.8 we model several *Rights* related to one *Resource*. For example, a resource *House* is associated with a right to use and right to sell. The type of the right (e.g. use or sell right) is modelled as a property *RightType* of a *Right*.

Rights to a resource need not be held by a single person or organization, and can be *partitioned* among several actors [Alchian and Demsetz, 1972]. For example, the owner of the house usually has a right to use and sell the house, but has no right to demolish it. Permission from the municipality is usually required to exercise this right.

[2] [Baida, 2006] also defines change of state resource, which represent a change of something, which can be an actor himself (e.g. a haircut), a physical good (e.g. cooking in a restaurant, shipping of goods), or information (e.g. book editing). In our ontology this is modelled with the Conversion activity.

3.5.3 Transfer and Conversion activities

A value transfer is valuable not only because of the rights and resources it transfers, but also because of the operational activities needed to transfer these rights and resources. Therefore, we model that every value transfer requires the execution of *operational activities*. For example, for a delivery service one should drive a car and upload the delivery. Operational activities are normally described in process models, e.g. in UML activity diagrams [Fowler and Scott, 2000]. They are also an important unit of analysis in the internal control theory, as discussed previously in section 2.1.1.

Note that the *operational activities* discussed here are *not* automatically *value activities*. A value activity is defined by [Gordijn, 2002] as a collection of operational activities, and, by definition, a value activity should be profitable for a performing stakeholder. This profitability requirement holds only for value activities, and not for operational activities which denote something to be done, in order to produce outputs as a result of inputs and resources, and is not required to be profitable.

There are two types of operational activities: *Transfer Activity* and *Conversion Activity*. A *Transfer Activity* is related to two kinds of transfers: giving a right for this resource and giving a resource in a physical custody. By the transfer a resource to someone's custody we mean to give the resource in that person's *possession* or to provide the person with *access* to the resource. In Figure 3.8 we model this distinction with a *Transfer Activity* that can transfer either a *Right* to a resource or a *Custody* of a resource. According to the property rights theory, having ownership rights to a resource and having it in custody are two different things. Ownership means the lawful possession of, or access to a resource, and may not come with actual physical custody. For instance, a person who steals a car possesses it, but has no legal rights to it.

Finally, the *Conversion Activity* models value transfers that require the conversion of a resource, e.g. a hairdresser services involves 'conversion' of a resource 'hair'. The *Conversion Activity* converts the *Resource*, not the *Value Object*.

3.5.4 Evidence documents

As we have already explained, one value transfer can be related to three types of operational activities: a *(Right) Transfer Activity* to transfer rights for a resource, a *(Custody) Transfer Activity* to transfer a resource in another actor's custody, and a *Conversion Activity* to change the state of a resource. Every activity can be complemented with an *evidence document* which contains information about the activity and the resources and actors involved in the transfer or conversion. For example, a receipt can serve as evidence of buying goods. So, in Figure 3.8, one *Activity* can be complemented with zero or more *Evidence Documents*.

In addition, an evidence document can be a part of a bundle of resources that represents a value object. For example, the value object *Stock* is of value to the owner, since it can be sold. The *stock certificate* is an evidence document, which certifies the transfer of the rights associated with the stock to the stock owner. The stock certificate is also often a required resource of the value object *Stock*, since without the certificate the owner cannot execute his rights (e.g. receive a dividend). In Figure 3.8 we generalize this finding and model that the *Evidence Document* represents a *Resource* of a *Value Object*.

3.6 Rights and evidence in *e^{3}value*

Since *e^{3}value* differs greatly from process modelling [Gordijn, 2002], it can be argued that a value transfer specifies a transfer of a right to a resource rather than a transfer of custody of a resource. In addition, representing transfers of evidence documents as value objects in value models may seem to be irrelevant, since evidence documents are often associated with procedures. On the other hand, as was already mentioned, there are examples, for which it makes sense to introduce value objects that represent transfer of custody. One such example is considered here and raises the question: Are transfers of custody and transfers of evidence documents *value* transfers?

We will now analyze an example which demonstrates that value transfers can be of value to actors not only when they transfer certain rights for resources, but also when they transfer the resources to the actor's custody. However, to represent an operational transfer of custody is *not a sufficient condition* for the transfer to be a *value* transfer. There should be a *willingness to exchange* this resource for another value object.

3.6.1 Rights and custody in one value transfer

In Figure 3.9 we model that someone buys a visit to a movie. We represent it with one value transfer between two actors - a cinema and a visitor. The visitor receives the value object *Movie Viewing*, and the cinema receives the value object *Money*.

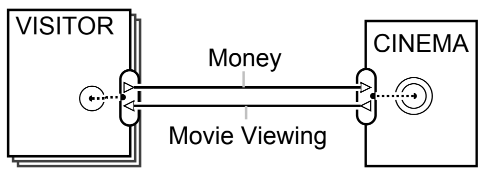


Figure 3.9: Example of visiting a cinema

Money is a value object that has a resource with the same name *Money*. Note that resources are not modelled explicitly in *e^{3}value* . We argue that the value transfer *Money* is associated with the operational transfers of both rights and custody of the resource *Money*. By paying money to a cinema, (1) the right to the money is transferred from the visitor to the cinema and (2) access to the money is also granted, since only then is it of value to the cinema. So, the transfer of custody is also required to gain value from a value transfer.

Movie Viewing is a value object that has many resources. For instance, a $Place$ in a cinema, and a $Film$. The resources $Place$ and $Film$ are not represented explicitly in the *e^{3}value* model. The value transfer *Movie Viewing* is similar to the value transfer *Money*, in that both are associated with the operational transfer of both value and custody of the resources $Place$ and $Film$. When money is paid, the visitor is granted the right to enter the cinema as well as the access to his seat $Place$ and to a movie $Film$. Access to the film and seat is of value to the visitor, since without them the visitor cannot view the movie and his ticket will expire.

So, we make two observations. Firstly, custody transfers can also be of value. For instance, access to a seat in a cinema is of value, since without it the visitor cannot view the movie and his ticket will expire. Secondly, a value transfer can be associated with both a transfer of rights and a transfer of custody of a resource on the operational level.

3.6.2 Splitting transfer of rights and custody

In some cases it can be useful to represent operational transfers of custody and rights to the same resource as separate value objects. In Figure 3.10 we show the same scenario as in Figure 3.9, but model the transfers of rights and custody to the resource $Film$ and $Place$ in separate value transfers. To do this, we explicitly model the ticket to the cinema as a value object.

Firstly, the visitor buys a ticket. This is modelled with the value transfers of *Ticket* and *Money* between the visitor and the cinema. The value object *Ticket* is associated with an operational activity of transferring the *rights* to the resources $Film$ and $Place$ to the visitor. The ticket certifies the transfer of rights and provides evidence that the visitor has a right to a certain seat and has a right to see a movie. In the ontology in Figure 3.8, the ticket represents an evidence document that certifies the *Transfer Activity*.

The value transfer *Ticket* models only the transfer of rights, but not the transfer of custody of the resources $Film$ and $Place$. If the visitor has the ticket, he either sells it to another visitor or exchanges it for access to the cinema. Only in the latter case is the custody of the resources $Film$ and $Place$ transferred to the visitor. This is represented with the value transfers *Movie Viewing* $_c$ exchanged for *Ticket*$_c$ between the visitor and the cinema (here the index C stands for 'custody'). The value transfer *Movie Viewing* $_c$ is associated with the custody transfer of the resources $Place$ and $Film$ to the visitor. In other words, it models that to gain full value of the exchange, the visitor must get access to the seat and see the movie. Similarly, the value transfer *Ticket*$_c$ is associated with the custody transfer of the resource $Ticket$ to the cinema. No rights are transferred by these two value transfers.

To summarize, as in the previous model, access to the resources $Place$ and $Film$ are of value because without the access the visitor cannot view the movie and his ticket will expire. However, in this case two different value transfers are being used to model the transfer of rights and the transfer of custody for the resources $Film$ and $Place$. *Ticket* value transfer corresponds to the transfer of rights to the resources, while *Movie Viewing* $_c$ value transfer corresponds to their custody transfer.

3.6.3 Criteria for distinguishing a value transfer

The questions that bothers many researchers (e.g. [Weigand et al., 2006, Kartseva et al., 2006a]) is whether it is acceptable to represent a transfer of an evidence document, like the *Ticket*, or a custody, like *Movie Viewing* $_c$, as value transfers. Are they really *value* transfers?

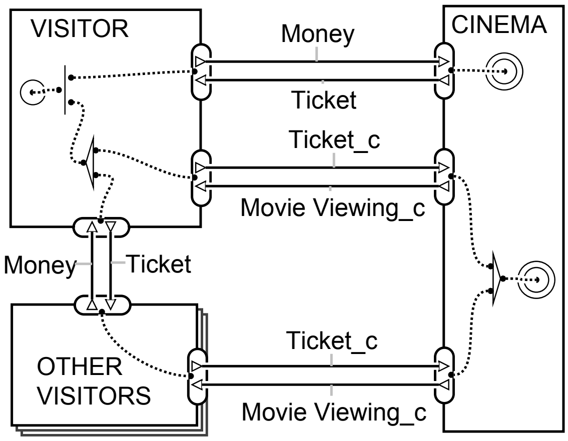


Figure 3.10: Example of a ticket modelled separately

The main criterion that is worth following when modelling an evidence document or a custody transfer as a value transfer is that the transferred object should be of value to some actor. In other words, at least one actor should be *willing to exchange* another value object in return for this object. An evidence document should only be modelled as a value object if it is of value to at least one actor. The same rule is valid for objects that represent only physical transfers.

In the example of the cinema, the value transfer *Movie Viewing* $_c$ is associated with a custody transfer for the film and the seat, but is not associated with transfer of any rights. However, as was previously argued, this is still a value transfer because without access the visitor cannot view the movie and his ticket will expire. Similarly, the value transfer $Ticket_c$ is associated with a custody transfer of the ticket, but is not associated with transfer of any rights. One of the reasons why possessing the ticket is of value to the cinema is that the cinema has to have an evidence of payments in order to ensure that money is not stolen by a cashier, tickets are not given away for free and seats are not given without tickets. Therefore, the ticket is of value due to control-related reasons.

To summarize, value objects can be of value to actors not only when the actors receive certain rights to the resources of these value objects, but also when they receive them in custody. However, representing an operational transfer of custody is *not a sufficient condition* for the transfer to be a *value* transfer. There should be another reason why having a resource in custody is of value to certain actors, e.g. because not having it in custody will lead to a value loss.

The choice of explicitly modelling transfers of custody and evidence documents as value transfers depends on the context and goals of modelling. Sometimes, modelling the provision of custody, evidence documents and rights is so trivial that it is not important to make them explicit in a model. In other cases, when the focus of the design is on rights, custody and evidence documents, modelling them explicitly is necessary to provide useful insights.

For example, it is only worthwhile explicitly modelling the ticket in the example of the cinema if we want to show that the visitor can resell the ticket. In Figure 3.9, where we do not consider the ticket as a value object, we are *not able* to model that the ticket can be resold.

3.7 Summary

In this section we have suggested the e^3*control* design framework and the e^3*control* ontology. Within the e^3*control* design framework we put forward a distinction between an *ideal situation* and a *sub-ideal situation* in networks. Whereas the ideal situation contains no unintentional errors or intentional irregularities, the sub-ideal situation does. Furthermore, the e^3*control* design framework suggests three steps in designing controls:

1. Designing the ideal situation;
2. Analysis of existing or possible sub-ideal situations;
3. Designing control mechanisms to reduce damage from the control problems.

The steps can be performed iteratively to handle multiple control problems. The framework is based on the internal control theory, as described in [Romney and Steinbart, 2006].

The e^3*control* ontology contains concepts that allow us to represent opportunistic behaviour in value models. It is an extension of the e^3*value* ontology of [Gordijn and Akkermans, 2003]. Among other things, the e^3*control* ontology includes such concepts as *sub-ideal value object* and *sub-ideal value transfer* in order to show that the result of sub-ideal behaviour is an incorrect exchange of a value object or an exchange of an inappropriate value object. An example of a sub-ideal value object is damaged goods. The transfers of sub-ideal value objects are marked in graphical models with *dashed lines*. In addition, we introduce a *liability token*, a concept that allows identification of a party who behaves sub-ideally.

Furthermore, we have also presented an extension of the e^3*value* ontology by linking the concepts of e^3*value* with a process model ontology. A value object is described by *Resources*, which are associated with *Rights* and *Custody*. A value port of a value transfer is associated with the operational activities of the actor who has this port. These activities are of two types: *Transfer Activity* and *Conversion Activity*. The *Transfer Activity* transfers a *Right* to a resource or a *Custody* of a *Resource*. The *Conversion Activity* is an activity that changes a *Resource*.

Finally, we have argued that transfers can be of value to actors not only when they correspond to transfers of certain rights to resources, but also when they transfer the resource's custody (e.g. granting access to a resource). However, to represent an operational transfer of custody is *not a sufficient condition* for the transfer to be a *value* transfer. In addition, we have argued that transfers of evidence documents can also be of value. For example, when an actor is accountable for something and when the evidence document provides the accountability. In general, the criterion for defining whether a transfer is a value transfer or not is that the transferred object should be of value to at least one actor in the model.

Chapter 4

Controls in the private sector

In this chapter we present a case study concerning the intellectual property rights clearance procedure for *webcasting*. This study demonstrates and explains e^3*control* in greater detail. Webcasting generally refers to the streaming of audio over the Internet and it is sometimes called "Internet radio". In order to broadcast music commercially, a radio station has to obtain several licenses and pay several copyright fees, including a fee for the *right to broadcast* music to the public. In Europe, this right is defined in European legislation[1]. A similar law, known as the Digital Millennium Copyright Act (DMCA[2]) exists in the USA . Each music track has several *rights owners*, who eventually receive the copyright fees. Songwriters and/or their music publishers, record companies that produce, manufacture and distribute records, and performing artists are examples of rights owners.

Radio stations have to pay copyright fees to many rights owners. Therefore, special organizations, called *rights societies* carry out the clearance procedure. These societies act as a 'middle man' between radio stations and rights owners. In Europe, Intellectual Property Rights (IPR) societies. Such societies operate on behalf of rights owners and provide radio stations with the service of clearing rights for a large group of rights owners, e.g. all Dutch rights owners. Although the rights clearance business is a government-regulated monopoly at the moment, a trend towards market liberalization can be discerned.

We commence by considering the model shown in Figure 4.1. It depicts individuals who wish to listen to music received from either an *ethereal radio station* or from an *internet radio station*. The listener decides which radio station to choose. This decision has been modelled by the OR-fork at the listener. For this case, we assume that only the value transfers related to Internet radio stations need to be considered. Consequently, we have not modelled any further exchanges between the ethereal station and other parties.

Two things must be done in order to listen to an Internet radio station. Firstly, the listener needs to secure an infrastructure, namely Internet access, of which various forms exist, e.g. broadband, fixed, or mobile. Secondly, the listener needs to hear at least one music track which

[1]EU Directive 2001/29/EC of the European Parliament and of the Council of 22 May 2001 on the harmonization of certain aspects of copyright and related rights in the information society. See EU Directive 2001/29/EC at www.europa.eu

[2]www.copyright.gov/legislation/dmca.pdf

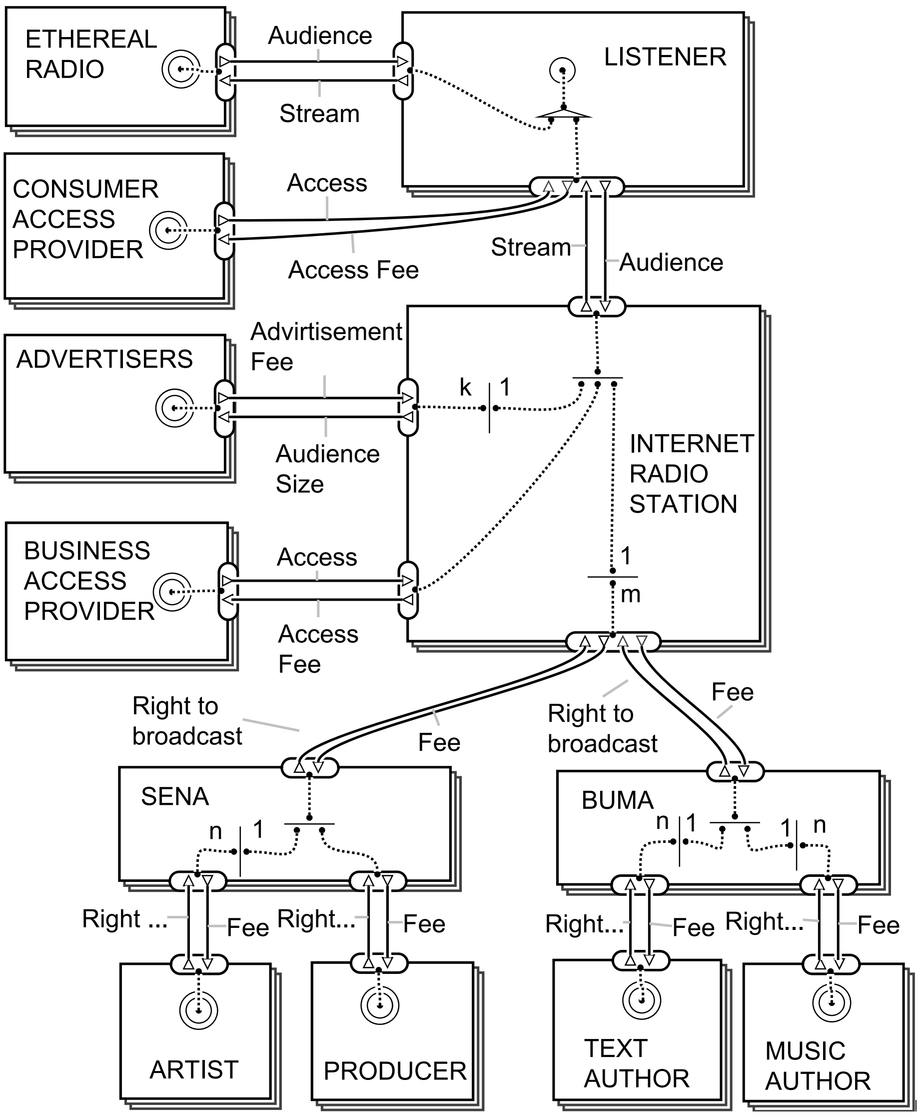


Figure 4.1: A Free Internet Radio business model

is broadcast by an Internet radio station. This is represented by a value object *Stream*, which is a stream of music tracks and in return for the stream, the Internet radio station gains *Audience Size*. The size of the audience is important to the radio station, because a large audience attracts advertisers, who are the main source of revenue for a radio station. The advertisers pay an *Advertisement Fee* for the audience size. In this model, we assume that listeners do not need to pay for listening: this is a free-to-air radio.

As we have explained, radio stations have to pay for the right to broadcast each track. The payment is made through rights societies, the most relevant of which in the Netherlands are: BUMA[3] and SENA[4]. Therefore, we have modelled value transfers of *Fee* and *Right to Broadcast* between the Internet radio station and SENA with BUMA. The rights are paid per track, not per stream. Therefore, there is an *explosion* element (see section 3.1) at the Internet radio station, which shows that a stream consists of *m* tracks, and for each track a right to broadcast has to be paid.

SENA and BUMA clear the same rights, but do so for different groups of rights owners. SENA transfers the money collected from the stations to artists and producers. BUMA does the same for music and text writers. Although there is only one producer per track, many artists, text writers and music writers are involved. Therefore, there are AND-forks and explosion elements, which show that *one* fee coming from the Internet radio station is distributed over *n* artists, text writers or music writers.

Each track is identified with the International Standard Recording Code (ISRC), a unique and permanent identification for a specific recording which in principle can be permanently encoded into a product as its digital fingerprint using watermarking technology. An encoded ISRC provides means to automatically identify recordings for the purpose of repartitioning rights payments. Radio stations pay a fee for the right to broadcast a track according to its ISRC. The fee is around E 0.0007 per track per listener.

At present, the information concerning tracks that are played is reported to rights societies by the radio stations themselves. According to www.riaa.com, every radio station has to report information about their broadcasting each calendar quarter. The problem with such a procedure is that a violation of the segregation of duties principle occurs (see section 2.1.1) when radio stations are allowed to report their own broadcasts. Radio station wish to pay lower rights-related fees, so, obviously, they have an incentive to *understate* either the number of listeners or the number of broadcasted tracks. Therefore, if a proper segregation of duties is to take place, the information about broadcasts should be provided by an actor who is not interested in reducing of the fees for the broadcast; this actor should be someone who is impartial.

There is no technology available for ethereal radio, which allows anybody other than a radio station to precisely identify what track is played and how many people listen to it. Therefore, market research is done on a regular basis to estimate the size of the audience. However, Internet does provide technology that enables other independent parties to calculate the exact number of listeners per broadcasted track as well as to identify the ISRC of the broadcasted track. In this case study, we have designed such a clearance procedure for webcasting taken from the value perspective using e^3*control* .

[3]See www.buma.nl

[4]In Dutch: De Stichting ter Exploitatie van Naburige Rechten. See www.sena.nl

4.1 Research Context

Our case study is based on information provided by SENA[5], an Intellectual Property Rights (IPR) society, responsible for collecting royalties for music broadcasting. J. Gordijn, a consultant at the SENA had access to the primary information within the realm of his work at the SENA. The author of this thesis acquired her knowledge of the case mostly from the information provided by J. Gordijn. In addition, she had access to the deliverables of the OBELIX research project (see http://obelix.e3value.com) and a series of articles, e.g. [Pedrinaci et al., 2005], where the case is described by other researchers.

4.2 Step 1. Ideal value model

The first step in designing controls is to construct an ideal value model. In fact, the model in Figure 4.1 is the ideal model of the rights clearance procedure. Note that, when specifying the ideal model here, we do not claim that regulations laid down by the Dutch rights societies are the best solution for Internet radio stations. The term ‘ideal value model’ simply indicates that this model shows that economic exchanges between different organizations are completely in accordance with the regulation of the Dutch rights societies.

For the remainder of this case study, we employ a more concise ideal value model, shown in Figure 4.2. We do so for reasons of brevity only. The differences are twofold. Firstly, the access providers and advertisers have been removed, because at this stage they play no role with respect to controls. Secondly, we have created a model for one rights society (e.g. one per country) and one rights owner (e.g. the main artist). This solution can be scaled up to a case with many rights owners and many rights societies. In addition, in Figure 4.2, the exchanges between the radio station and rights societies are on a *per track per listener* basis. Therefore, we have modelled a value object *Track* instead of *Stream* and *Listener* instead of *Audience*. So, in order to calculate how many people have listened to a specific track, we need to know the number of listeners in a certain market segment. In order to determine how many tracks were played, we then have to calculate the number of executions of the consumer need element in the e^3*value* model.

4.3 Step 2. Sub-ideal value model

In the ideal model in Figure 4.2 we assume that the information about actual broadcasting is delivered to the rights societies by the Internet radio station. This information is *trustworthy* only if we assume that the Internet radio station reports the tracks they have played correctly. However, in a sub-ideal situation both Internet radio stations as well as rights societies may misrepresent information, which results in a loss of revenue for rights owners. In addition, administrative errors can be made which also result in the loss of the revenues. Such situations are considered here as sub-ideal.

[5]In Dutch: De Stichting ter Exploitatie van Naburige Rechten. See www.sena.nl

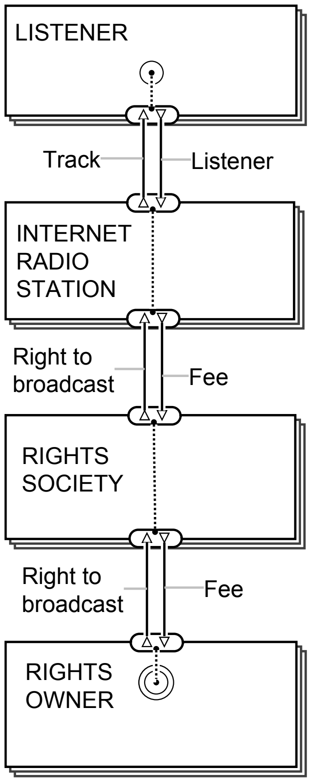


Figure 4.2: An ideal value model for Internet radio business model

Figure 4.3 presents the various sub-ideal paths that reveal control problems in a situation where one of the actors behaves sub-ideally. We assume that both the Internet radio station and rights society will expose sub-ideal behaviour. The listener has no prime interest in doing so, because listening to music is free anyway.

In Figure 4.3 each track is marked with the letters A, B or C, which correspond to the ISRC code for a track. As was explained in the case description, the ISRC code is a unique identifier of every track. Henceforth, *Track A* stands for a broadcasted track (also exchanged between the listener and the Internet radio station), and other labels, B and C, refer to other tracks than the broadcasted *Track* A. This distinction is required to show that someone could pay for the *wrong* track. In addition, the rights owners of a track are also labelled with the same track ISRC (A, B or C). This indicates that every track is associated with a specific rights owner.

As explained in section 3.3.3, a sub-ideal value object can be *empty* or *incorrect*. Based on this, we identify two kinds of sub-ideality for the value object *Right* (short for *Right to Broadcast*) and *Fee*: (1) the rights to play a track to a specific listener are not paid for at all or (2) the rights to play a track are paid for the wrong track (e.g. *Track B* instead of the played *Track A*). In both cases, the authentic rights owner of the played track is not paid. In addition, in the second case, the wrong rights owner receives the money (e.g. a party owning the rights for *Track B*, not the played *Track A*).

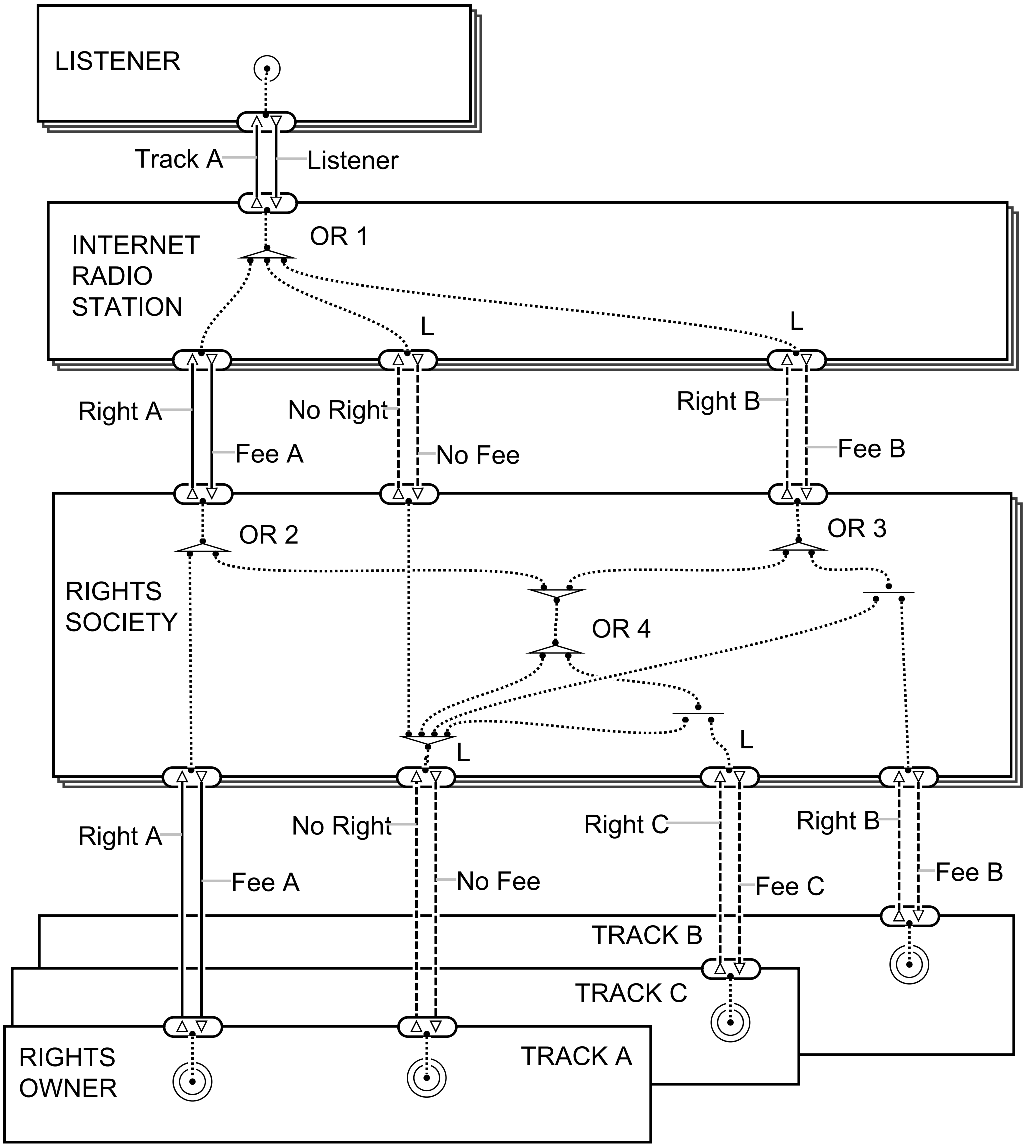


Figure 4.3: A sub-ideal value model for Internet radio business model. In this model we refer to 'Right to Broadcast' as 'Right'

Internet radio station. Figure 4.3 shows how an Internet radio station can behave either ideally or sub-ideally. This choice is modelled with the *OR-1* fork at the Internet radio station. In the ideal situation, the ideal value transfers are the same as in the ideal model: *Fee* A and *Right A*, where A is an ISRC of the broadcasted track. In the sub-ideal situation, the Internet radio station can do two things: (1) it does not pay at all for the broadcasted track, and (2) it pays for a wrong track. The first sub-ideal situation (not paying for the track at all) is modelled with the "dashed" sub-ideal value transfer *No Fee*. As a result, no right is issued, because in a situation with no controls the rights society does not know that the Internet radio station is cheating, so it just assumes that no track was played. However, the absence of the rights does not stop the radio station from broadcasting this track: the transfer of *Track A* between the listener and the radio is always ideal, even though the transfer of the rights between the radio and the rights society may not be. The second sub-ideal situation (buying the right for another track, e.g. *Track* B) results in the sub-ideal exchange of *Right B* in return for a sub-ideal *Fee B*.

The two sub-ideal value transfers *No Right* and *Right B* between the Internet radio station and the rights societies are the responsibility of the Internet radio station. To represent this, we use two liability tokens **L** at the Internet radio station actor.

Rights society. Sub-ideal behaviour relates to a specific actor. So, even though an Internet radio station behaves ideally, the rights society may not. This choice is modelled by a set of OR-forks at the rights society.

If the Internet radio station behaves ideally and exchanges *Fee* A and *Right A*, the path leads to an *OR-2* fork. At the *OR-2* fork the rights society has a choice to behave ideally or sub-ideally. If the rights society behaves *ideally*, the *OR-2* fork leads to the ideal value exchange of Right A and *Fee* A between the rights society and the rights owner of the *Track A*. If the rights society behaves *sub-ideally*, the *OR-2* fork leads to another *OR-4* fork, where it again has again a choice of two sub-ideal paths, similar to those at the Internet radio station. If the rights society does not pay, the path at the *OR-4* fork leads to the exchange of *No Right* and *No Fee*. If the rights society pays for a wrong track, the path leads to payment to a wrong track owner (see *Right C* and *Fee* C) as well as to non payment to the authentic track owner (see the AND-fork leading to exchange of *No Right* and *No Fee*).

If the Internet radio station does not pay, neither does the rights society, since it does not know that the track was played. Therefore, the exchange of *No Right/ No Fee* between the radio and the rights society leads to the exchange of *No Right/ No Fee* between the rights society and the rights owner.

If the Internet radio station pays for the unplayed *Track B*, the rights society may also choose to behave ideally or sub-ideally at the *OR-3* fork. If the rights society behaves sub-ideally at OR-3, it pays for *Track* C instead of *Track* B or does not pay at all. These choices are modelled with the fork OR-4, previously explained.

If the rights society behaves ideally at *OR-3*, it will pay for *Track* B. This path leads to paying *Fee B* to the wrong rights owner of *Track B* and (see the AND-fork) paying nothing (*No Fee*) to the authentic rights owner of *Track A*. In this case, the ideal behaviour of the rights society still results in sub-ideal value transfers (*Fee B* and *No Fee*). The rights society does not know that *Track B* has been reported incorrectly, so they assume that paying for *Track B* is ideal.

So, wrongdoing is committed by the Internet radio station and not by the rights society. This difference can be seen from the liability tokens, which in this case are assigned to the Internet radio station, and not to the rights society.

4.4 Step 3. Modelling controls

In the previous section we have identified several control problems. The first one arises when either the Internet radio station or the rights societies, or indeed both parties do not pay at all for a track broadcast. The second occurs when the parties pay for a wrong track broadcast. The object of this section is to design controls to mitigate the consequences of such sub-ideal behaviour.

4.4.1 Solution 1: Collecting evidence from listeners

A control mechanism may address a sub-ideal path in two ways: firstly, it may detect a sub-ideal path execution, secondly, it may prevent a sub-ideal path execution. In this section we focus on modelling a specific *detective control mechanism* that can be used to assess whether all tracks are cleared or not.

Controls against problems caused by an Internet radio station

Firstly we assume that only an Internet radio station behaves sub-ideally. The described control problems described are caused by the transfer of untrustworthy information about the broadcasted tracks and the number of listeners. To prevent this, the rights society should find a trustworthy source of information about the broadcasts. One solution which is possible on Internet, is to collect this information from listeners. If every listener shares information about each track listened to, the rights society would be able to count the number of rights required per listener per track without consulting the Internet radio station.

However, it is most unlikely that many listeners would be willing to share information, due, for instance, to concerns about privacy. From the e^3*control* point of view, they have to get something 'in return' to be willing to share the information. One way is to pay the listeners for sharing the information. Another way is to allow the listeners to listen to a music track only if they agree to share this information. In value models, this can be modelled by introducing the value object *Right to Listen*. A right to listen is granted to listeners in return for sharing information about tracks they listen to. A right to listen should be obtained to ensure that every listener really shares the information.

To model this in e^3*control*, we have added two value objects to Figure 4.4, exchanged between the rights society and the listener. The first value object is the *ISRC code*, which gives information about the track, listened to by a particular listener. The ISRC code is a value object: it is of value to the rights society, because without the knowing the exact code the rights societies cannot execute their core business of charging the correct and complete fee for broadcasting. Essentially, the rights society 'buys' this information from the listener.

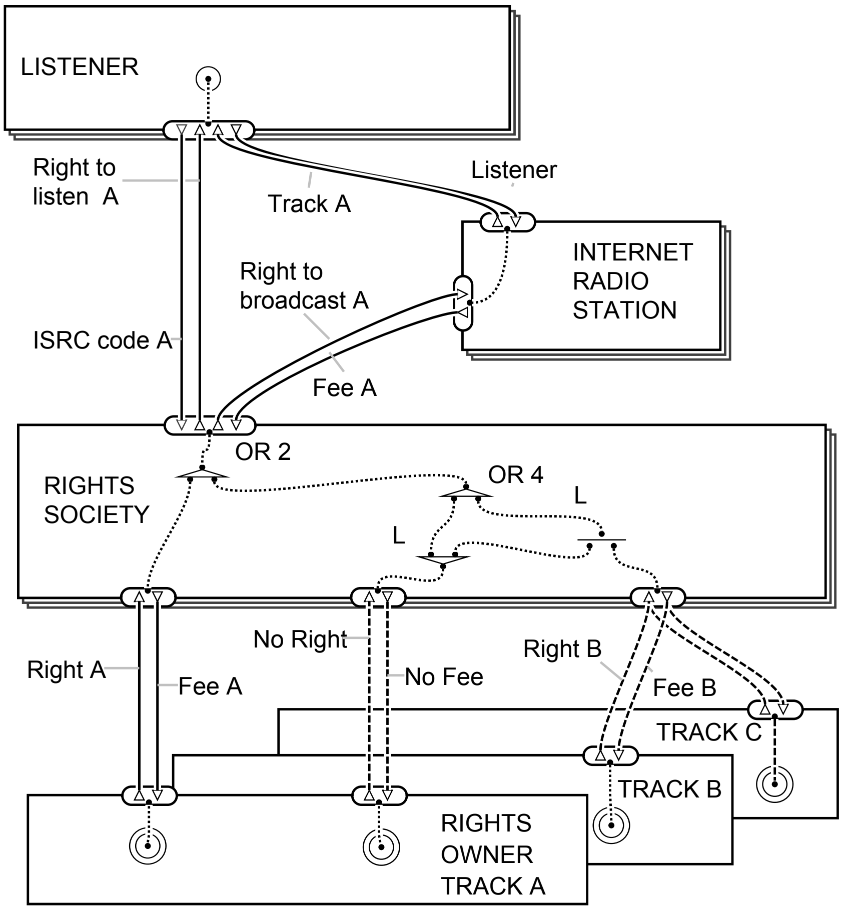


Figure 4.4: Control of the Internet radio station

The second value object is the *Right to Listen* transferred from rights societies to listeners. Only after receiving the right to listen should the listener be allowed to listen to the music. A listener has the value object *Right to Listen* in the same interface as the value object *ISRC code*. This shows that the listener does not get the right to listen without sharing the information about the tracks listened to. Furthermore, the value object *Right to Listen* is also in the same interface as the value objects *Track*. This model shows that the listener cannot listen to the track without having the right to listen. As a result, the listener is not able to listen to an Internet radio station unless he shares the information about the ISRC code. This is different from the right to broadcast, where an Internet radio station has the object *Right to Broadcast* in a different interface than the object *Track*, which means that it is possible to broadcast the track without having the right to broadcast it.

Modelling reconciliation in e^3*control*. The e^3*control* model also represents a reconciliation aspect of control. The notion of a value interface requires that the number of value objects, exchanged through it, is the same. That is to say, at the value interface of the rights society, the number of objects *Right to Broadcast* issued to the Internet radio station must be the same as the number of objects *Right to Listen* and *ISRC code* requested by listeners, and the same as the number of object Fee delivered to the rights society.

Since the right to listen is only issued if track information is exchanged, the number of objects of *Right to Listen* and *ISRC code* is always the same. The *Right to Broadcast* is issued as soon as the *Fee* is paid, so their number is also always the same. However, if the Internet radio station lies about the number of listeners, for instance, it reports 10 listeners instead of 100, and does not pay a fee for all the listeners, then an inconsistency at the value interface will be the immediate result. The e^3*value* tool generates an error message if we assign 1000 exchanges of the *Right to Listen* and only 100 exchanges of the *Right to Broadcast*.

If the internet radio reports *Track B* instead of *Track A*, an inconsistency at the value interface will also be visible, because *Right to Listen* and *Right to Broadcast* have to have to refer to the same ISRC. Thus, reconciliation should reveal that the right to listen is issued for a track with the *ISRC code A*, while the right to broadcast is issued for a track with the *ISRC code B*. This reconciliation has not yet been implemented in e^3*control* .

Controls against problems caused by rights societies

Assuming that the exchange of the right to listen to a specific track is guaranteed, the sub-ideal exchanges *No Right/ No Fee* and *Right B/ Fee B* between an Internet radio station and a rights society are detectable, and therefore they have been removed from Figure 4.4. However, other sub-ideal exchanges caused by the sub-ideal behaviour of a rights society still remain and are not targeted by the controls that have been introduced.

To eliminate the remaining sub-ideal exchanges, reconciliation should be executed by a party other than the rights society. Ideally, the remaining party that can provide the right to listen is the rights owner. In Figure 4.5 the right to listen has been issued to the listener by the rights owner. Reconciliation of this right to broadcast can now be performed at the four-port value interface of the rights owner. So, the number of exchanged rights to listen to a specific track should be equal to the number of rights to make it public.

Additionally, it is unrealistic that every rights owner, be it an artist, a musician or a text writer, set up their own rights management. Therefore, we now introduce a new actor *rights manager* who performs the technical rights management on behalf of the rights owner. Actually, the rights manager has to guarantee the exchange of the right to listen. This is done at operational level and is described in the next section.

4.4.2 Solution 1: Information systems viewpoint

In the previous section, we have proposed adding an additional right: the right to listen to a music track. So, the listener should be able to obtain both this right and the requested stream of tracks. This is expressed by the value interface of the listener actor. We can guarantee that the ideal semantics of the interface (exchange all objects, or none at all) hold by using encryption technology. One such technical solution is modelled here.

The conceptual model of the solution is shown in Figure 4.6. Note that this figure is not an e^3*control* model, rather it is more similar to a UML collaboration diagram

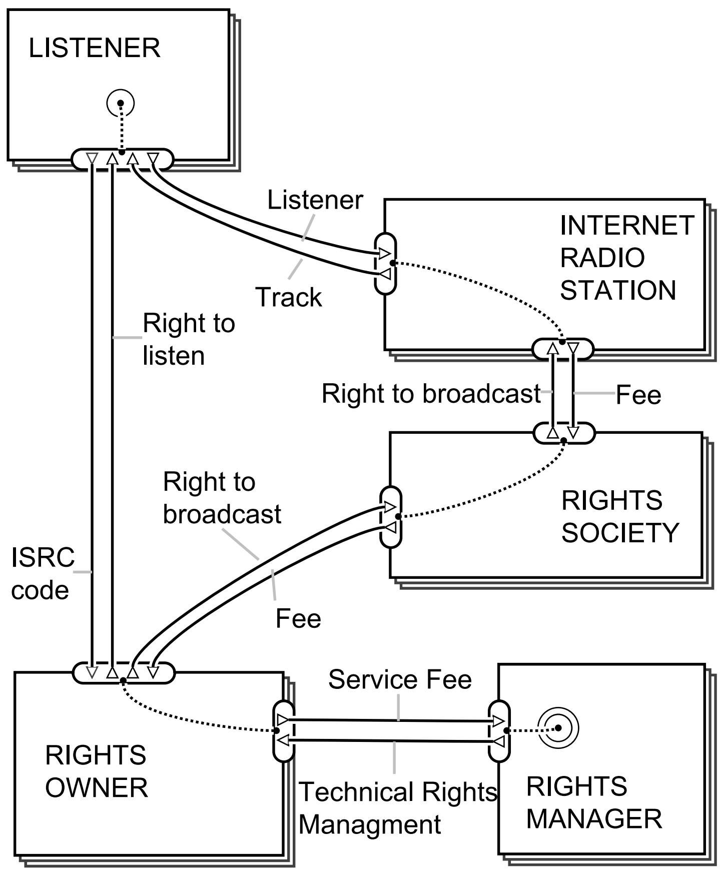


Figure 4.5: Introducing a trusted third party to control both Internet radio station and rights societies

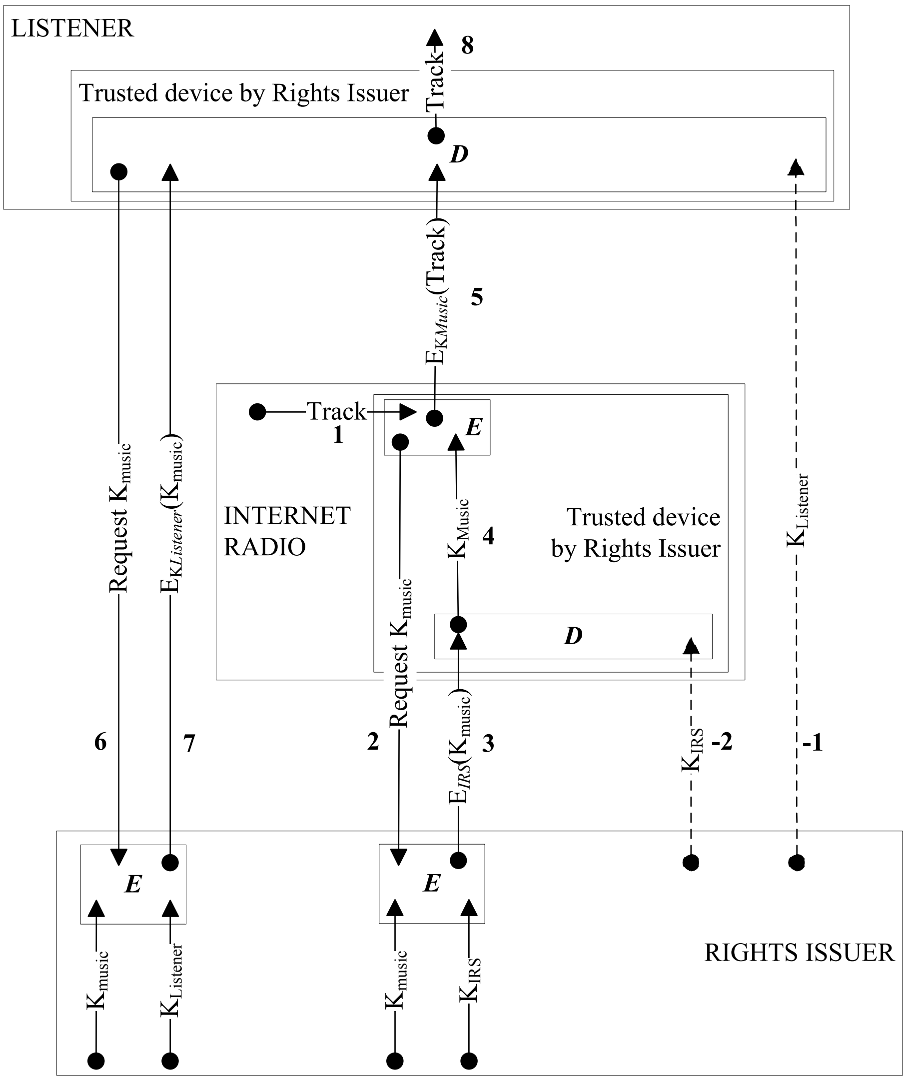


Figure 4.6: Using encryption technology for track counting

[Rumbaugh et al., 1999]. Arrows indicate messages that are exchanged between actors (boxes). Numbers next to the arrows indicate a time sequence. Boxes with an "E" denote an encryption operation, whereas "D" stands for decryption. This solution 'translates' the rights to a cryptographic key issued to the various parties. We distinguish three parties: the listener, the Internet radio station (as in the value model) and the rights issuer. The rights issuer can be a rights society, a rights owner, or an organization operating on behalf of these e.g. rights managers).

The rights issuer and the Internet radio station have agreed in advance on an encryption key, K_{IRS} (message -2). The same holds for the rights issuer and the listener, they agreed on $K_{Listener}$ (-1). How these keys are exchanged does not fall within the scope of this work, but one possibility is that these keys are stored on a smartcard which is issued by the rights issuer to the listener and the Internet radio station, respectively. The assumption that these keys are distributed in advance is denoted by the "-" sign in the figure.

Both the Internet radio station and the listener have a so-called *secure computing and storage device* at their premises. Such a device is tamper-proof and is trusted by the rights issuer. In practice, a secure device may take the form of a smartcard, but it is also possible to implement such a device in a software component (however, in general a hardware smartcard is more tamper-proof than just software code). It is important that the listener and the Internet radio station do not have access to this device (without damaging it). The Internet radio should not be able to modify the information on played tracks. Although listeners are not interested in modifying the information on listened to tracks, they need protection at their site, since the Internet radio can easily play the role of the listener. In addition, the protection at the listener site constitutes a situation of non-free radio. Therefore, the keys, K_{IRS} and $K_{Listener}$ are stored on the secure devices of respectively the Internet radio station and the listener. So, although the Internet radio station and the listener physically have the keys, they cannot read them because they are stored inside a tamper-proof device, which they can not access.

If an Internet radio station broadcasts a track (1), its secure device first requests a key, K_{music} (2) This key is used later on to encrypt a music track in such a way that to listen to it, a listener must obtain a key to decrypt the track. This K_{music} is issued by the rights issuer and encrypted with K_{IRS} (3), a secret shared by the rights issuer and the Internet radio station. Consequently, no one can read K_{music}, not even the Internet radio station, since the key is on a secure device, not accessible by the radio station. The encrypted music key (denoted by $E_{K_{IRS}}(K_{music})$) is decrypted by the Internet radio station's secure device (denoted by D), resulting in the plain key K_{music} (4). This key is used by the secure device of the Internet radio station to encrypt the track ($E_{K_{music}}$(Track)) (5). Finally, this encrypted track is broadcasted and received by each listener.

Before the track can be listened to, the listener's secure device has to decrypt the encrypted track by obtaining K_{music}. So, the device requests this key from the rights issuer (6) who logs the request for counting purposes. The rights issuer compares the number of requests with the number of tracks reported by the Internet radio station. He then sends the music key, encrypted with an earlier agreed upon listener's key ($E_{K_{Listener}}(K_{music})$) (7). The listener's secure device decrypts this message and uses the obtained key to decrypt the track and finally plays it (8).

The information system solution implements some aspects of the control mechanism designed at value level in Figure 4.5. More about this can be found in the discussion section.

4.4.3 Solution 2: Sharing information with advertisers

Besides the controls described in the previous two sections, there are other ways to ensure the authenticity of the information reported by the Internet radio station. In this section we look at one such a mechanism, which has a more preventative nature.

In this control, we suggest that the rights society should check if the number of listeners reported by the Internet radio station is the same as the number of listeners reported to *advertisers*. The fee paid by advertisers is also a function of the number of listeners: the bigger the radio audience, the higher the fee they can charge the advertisers. It is therefore in the interest of an Internet radio station to increase the fee paid by advertisers. If advertisers and rights societies

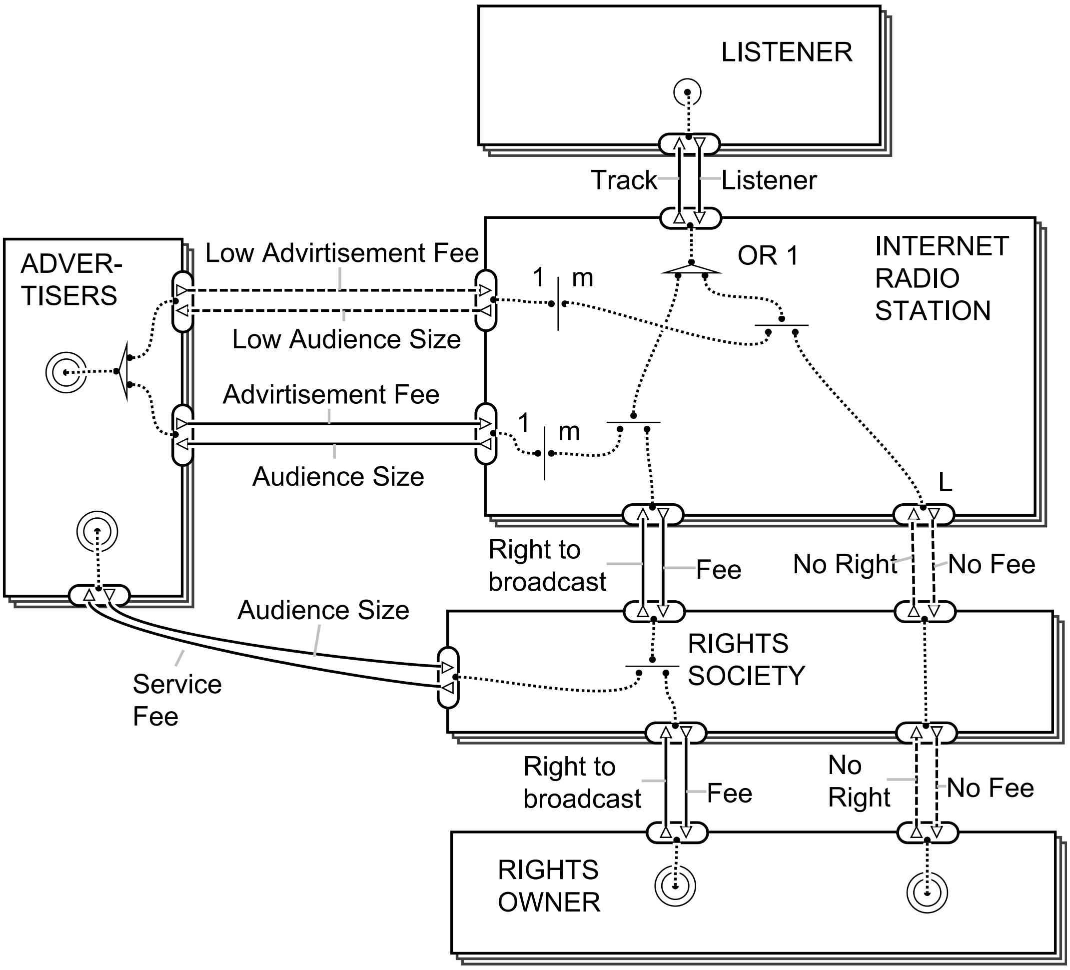


Figure 4.7: A solution involving a control service provided by advertisers

share information about the number of listeners, the Internet radio station would not be willing to understate their number, since it would mean a decrease in advertising fees.

A value model with controls. In Figure 4.7 we show a value model for this solution. The underlying assumption is that the Internet radio station first reports the information to advertisers who then report it to the rights societies. Another assumption is that the advertisement fee is charged on a *per listener* basis.

In Figure 4.7 we show the advertisers who pay the Internet radio station an *Advertisement Fee* and get the *Audience Size* in return. The *Audience Size* gives information about the size of the radio station's audience of course. In fact, this is a number of listeners who listened to the radio in a certain period of time, e.g. daily. Because we assume the exchanges per listener per track (unlike in Figure 4.1) we introduce an implosion element. This element indicates that per *m* listeners one object *Audience Size* is exchanged to the advertiser.

As in the sub-ideal model in Figure 4.3, we assume at the *OR-1* fork that the Internet radio station either reports the track listened to by the listener and pays the rights, which corresponds to the value transfers of *Right* and *Fee*; or the radio does not report the listener and pays no fee for the rights, which corresponds to the value transfers of *No Right* and *No Fee*. We do not consider other sub-ideal exchanges from Figure 4.3 in this model, since this solution does not mitigate them, as will be explained later in the discussion section.

We assume that the information about the audience size has been shared by the rights society and the advertisers. Therefore, if the Internet radio station pays the rights fees for a lower number of listeners, the advertisement fee will also be lower, since it is based on the lower audience size which is taken as an estimation of the rights fees. The understated audience size and the lower advertisement fee are modelled by the value objects *Low Advertisement Fee* and *Low Audience Size*. To model the dependency between the fee for the rights and the fee for the advertisements in e^3*value* , the value transfers *No Fee* and *Low Advertisement Fee* are connected by an AND-fork. Because they are sub-ideal, *Low Advertisement Fee* and *Low Audience Size* are marked with a dash similar to *No Fee* and *No Right*.

In this model, the Internet radio station is not interested in playing down the number of listeners to the rights society. Should it do so, then this information will be immediately shared with the advertisers, and will result in a lower advertisement fee. Thus, what the station gains on the lower rights fees, it will lose on lower advertisement fees. For this reason, the Internet radio station should prefer the ideal path of reporting the correct number of listeners rather than a lower one.

In this model we show that the Internet radio station sells *Audience Size* to rights societies. It is realistic to assume that rights societies are interested in this information and are ready to pay for it, since this information is trustworthy. Such trustworthy information about the number of listeners is valued by rights societies, since it is crucial to their core business.

Size of *Advertisement Fee* and *Rights Fee*. For this system to work, the value the Internet radio station gets when reporting a listener and paying for the rights must be higher than the value it gets when not reporting a listener and not paying for the rights. In other words:

$$AdvertisementFee - (Rights)Fee > NoAdvertisementFee - No(Rights)Fee.$$

Since the right part of the inequality is always zero, this condition states that the amount the radio pays for the rights should be less than the amount it gets from advertisers. Otherwise, if the radio gets fewer advertisement fees than the fees it pays for the rights, it will be tempted to play down the number of listeners. On the other hand, under such conditions the Internet radio station has a negative cash flow, so it is better for it to stop executing the business model.

4.5 Discussion

Discussion of the first solution Market liberalization means that new rights societies and rights managers may emerge, that radio stations will be no longer obliged to clear rights with a specific rights society, like BUMA and SENA, and that rights societies will compete in representing artists and other rights owners. The new value model in Figure 4.5 is in line with the EU policy towards liberalization of the rights management market [Pedrinaci et al., 2005], since it can be adjusted to include participation of several rights societies.

The information system solution does not yet implement all aspects of the control mechanism designed at value level in Figure 4.5. That is because K_{music} is not bound to a specific track (modelled with labels A, B etc.), so this solution cannot guarantee that the correct track is cleared, it only checks that a track is cleared. So the Internet radio station can still follow the sub-ideal path along which it buys the rights for a wrong track (see *Right B* and *Right C* in Figure 4.3).

How can we prevent an Internet radio station from combining a series of tracks into one track and offering this one combined track to its secure device for encryption? There are some solutions to this problem. First of all, the secure device can intelligently detect change of tracks. Such technology is successfully used, e.g. to remove commercials from a video stream [Hanpachern, 1982]. Secondly, the rights issuer's computer can listen to a stream of tracks broadcasted by the Internet radio station and carry out intelligent track detection. The detected tracks can then be compared to the reported tracks. By using time-stamps, detected tracks can be bound to the logged tracks per listener.

Discussion of the second solution A disadvantage of the second solution is that it only provides trustworthy information about the number of listeners, but not about the track played. In addition, it only takes the sub-ideal behaviour of the Internet radio station into consideration, but not that of the rights societies. As a result only one sub-ideal path of the sub-ideal model in Figure 4.3 is mitigated: the exchange of *No Right* and *No Fee* by the Internet radio station. The risks described by the other paths are not mitigated.

Another disadvantage is that the Internet radio station might be willing to overstate the number of listeners, if the fees collected from advertisers are much higher than the fees paid for the rights. This can be done, for example, with a purpose of generating a higher turnover. To avoid this, another control should be introduced to ensure that the radio station does not overstate the number of listeners, e.g. by taking samples of the audience density from time to time. However, this is not really a risk for the rights owner as he simply acquires more money.

Although this solution is not the most effective, we have included it here for the reasons of illustration. Firstly, it shows that there can be many solutions to one problem. Secondly, it has several interesting aspects. This solution introduces a kind of an *incentive* for the Internet radio station not to cheat. The incentive is based on the fact that if the Internet radio cheats, it will lose a revenue. The incentive not to cheat is the difference in the advertisement fee and rights fee. This difference should stay positive to ensure that the radio station does not understate the number of listeners.

This is an example of the incentive control, discussed in section 2.1.2. Furthermore, the solution is interesting, because it is based on a complex conflict of interests between the three parties: the Internet radio station, the rights societies and the advertisers. Such conflict of interest is possible only if the advertisers are involved and if the advertisement fee is a function of the same variable (number of listeners) as the rights fee.

4.6 Lessons learned

In the summary of lessons learned we reflect on our expectations about value modelling for controls and compare them to the results of this case study. We also discuss the expressive power of the e^3*control* ontology, as demonstrated in this case study.

Lesson 1. Controls as commercial services with added value. This case study demonstrates that controls can be commercial services. One such service observed is provisioning of the ISRC code by the radio listener to the right society. Another control service is the technical rights management, provided by the rights issuer to the rights owner. As argued below, the ISRC code is an evidence document, which is of value.

Taken that the controls are seen as services, modelling them from the economic value perspective reveals different aspects of controls, as compared to the process of information system models. Take for example the first control service. The listeners provide the ISRC code information for free. At the first glance, this means that the listeners do not get anything in return for providing the information. However, this is not the case. In terms of e^3*value* , nothing is exchanged for free and each economic exchange should have a corresponding reciprocal exchange. This way of thinking forced us to think about what the listener gets in return for providing the information, which, we have argued, is the *right to listen*. Indeed, the listeners can only listen to a radio if they have access to the music. In paid radio stations, the listeners buy this access, which can be interpreted as buying the right to listen. However, if the access is not restricted, as in this case study, the right to listen is granted automatically for free. In this model, we suggest that the listener should be granted the right to listen in return for providing the information about the listened to tracks.

Realizing that the right to listen exists and modelling it explicitly with e^3*control* is important, because it allows us to interpret the underlying information system solution as a mechanism that guards the exchange of the right to listen. To summarize, the e^3*value* requires modelling the reciprocal exchanges for control services, which leads to a a better understanding of the control mechanism.

Lesson 2. Value aspects of control instruments. We have observed several control instruments in this case study, both rights and evidence documents. The reason of modelling evidence documents as value objects is because they carry information which is of value. In the case of music, the *ISRC code* is an evidence document, which is modelled as a value object because the information it represents is of value to rights societies. Without having reliable information about broadcasted tracks and audience size, rights societies cannot execute their core business accurately: charging radio stations a fee *per listener per track*. This way of thinking goes along with our argument in Chapter 2 that information-related objects in value models are value objects as long as they are of *value* to some actors.

Another control instrument with a value aspect is a *right*. Rights are generally considered to have a regulatory purpose, and, ate therefore control instruments. Two rights are defined in this case study: *Right to Broadcast* and *Right to Listen*. Both rights are modelled as value objects, since they carry economic value. *Right to Broadcast* is of value to rights owners, since they receive a fee for this right. *Right to Listen* is of value to the rights society or rights issuer, since they receive an information about the ISRC code in return, which is a value object, as have been discussed above.

Lesson 3. The value viewpoint is an abstract rationale for procedural aspects of controls. We have shown how to model controls from two viewpoints: (1) the value viewpoints and (2) the information systems viewpoints, in this case in terms of a cryptographic implementation. We consider it to be important to distinguish the two viewpoints, since both of them focus on different issues: issues of a business model and issues of technical implementation. The advantage of the e^3*control* models is that they allow us to quickly sketch the control problems by using such concepts as *sub-ideal value object*, *liability token* and others. The analysis is very quick, since we abstract from any procedural details. Another advantage of e^3*control* models is that they identify a number of control problems and define mechanisms to prevent or detect them. Information system models are not particularly developed to analyze the control problems; they are primarily being used to describe a solution.

We have seen in the case study that the high-level economic value analysis guides the design of the information system solution. The information system solution can be seen as a solution that ensures occurrence of all value transfers of the corresponding value model. From this point of view, the preliminary value modelling is important, because different economic value solutions require different operational models to implement it. For instance, an information system required to implement a business model for a paid radio is different from the considered in this case business model for the free radio.

Furthermore, the economic value model provides criteria to judge whether an information systems solution is sufficiently good to mitigate the identified control problems. As seen in this case study, the information system solution requires some adjustments to mitigate all the control problems; however, a less simple solution could be provided to mitigate some of the identified problems. A further value-level analysis could be done to identify what are the costs of the adjustments and if they are justified by the benefits. This analysis could not be done if we limit our analysis to the information system perspective.

Lesson 4. The e^3*control* ontology. First of all, this case study demonstrates that modelling controls with value models is possible. Several control problems have been identified at the start of the case study and, as a result of the analysis, all of them could be modelled using the e^3*control* concepts. In general, all e^3*control* concepts have been useful in describing the control problems we have encountered. Such concepts as sub-ideal value object and sub-ideal path provide means to model control problems.

We have also managed to create models which comply with the e^3*value* ontology is the sense that they do not contain loops in the dependency paths. Neither do the ideal value models contain one-way value transfers.

There are several aspects in the e^3*control* ontology we would like to discuss within the realm of this case study:

Separate transfers of rights and custody. In Chapter 3 we have argued that it is important to understand what corresponds to a value transfer on the operational level: a transfer of a resource's *custody*, its *rights* or both. We have also argued that the transfer of custody and a transfer of its rights can correspond to different value transfers in the value models. This case study confirms that in some cases it is necessary to model a transfer of a resource's rights and transfer of a resource's custody as separate value transfers. The value object *Right to Broadcast* and *Right to Listen* represent rights associated with a resource *Track*. However, access to the resource *Track* and transfer of the right to broadcast the track are modelled as separate value objects. The transfer of custody for the resource *Track* is modelled in the value transfer of the value object *Track* between the listener and the radio. The rights to broadcast the track and the right to listen to the track are modelled by two other value object *Right to Broadcast* and *Right to Listen*.

Modelling of reconciliation with a value interface. The value interface concept of e^3*value* provides formalism to model reconciliation of two or more value objects. As such, we were able to generate an error message with the e^3*value* software tool (see www.e3value.com) if a value interface contained different amounts of value objects. It is, however, not possible to reason about reconciliation of properties of the value objects. In sum, the value interface concepts allows us to model reconciliation controls.

Incentives as controls. An e^3*control* model allows us to represent economic incentives as was demonstrated by the second solution. The incentive for the Internet radio station not to cheat depends on the difference of the advertisement fee the radio receives and the rights fee the radio pays. This difference should stay positive to ensure that the Internet radio station does not understate the number of listeners to avoid the rights fees. Incentives are contractual controls, as discussed in the section 2.1.2.

4.7 Summary

In this section we have described a case study concerning a music rights clearance procedure for Internet broadcasting. We have used e^3*control* to analyze control problems and suggested solutions to (possible) understatements of actual broadcasting information by Internet radio stations and rights societies. In addition, we have also put forward an information-system level solution.

The goal of the case study has been to demonstrate that value modelling can be carried out for a high-level analysis of controls in networks. Firstly, we are able to model reconciliation mechanisms by employing the concept of value interface. In addition to that, we are able to represent controls as services and see how controls change the business model of the network. In addition, value modelling facilitates the understanding and design of controls based on financial incentives. The added value of the value modelling approach, as compared to traditional process modelling, is that it focuses on aspects of value, even when considering exchanges of information.

Chapter 5

Controls in the public sector

Inter-organizational controls often occur in highly regulated sectors, such as the electricity, telecommunication and health care sectors. In such regulated environments, production, distribution, and pricing decisions are made by a regulatory authority rather than defined by market forces. A reasonable question is then whether the e^3*control* approach, which relies on e^3*value* and its notion of the reciprocal economic exchange, make sense in such situations. To investigate this question we have carried out a case study in the health care sector, specifically on recent changes in the governance and control of a public insurance system in the Netherlands. Another interesting aspect of this case is investigation of how controls are used in a non-commercial network setting.

In the Netherlands, the system for exceptional health care deals with long-term and chronic diseases, such as protracted illness, invalidity, learning disability, mental disorders and geriatric diseases. An example of such illness is dementia. Because this kind of care is too expensive to be covered by every ill person individually, in the Netherlands the system is arranged as *public* health care insurance. The AWBZ[1], an Act on Extraordinary Medical Expenses, regulates the provision of this area of care. A patient only pays a small part of the costs; the largest part of the costs is reimbursed to the care provider from a government fund, collected from taxes. Currently, the exceptional health care system is undergoing a major change in terms of services, finances, actors, and controls.

Clearly, the AWBZ system qualifies as a highly regulated environment. Patients need to apply for care through a cumbersome bureaucratic process. Various semi-independent governmental agencies perform tasks in the system, in return for government funding. Although the government retains political control over the system, increasingly also private care providers are allowed to enter.

The goal of this case study is to design existing controls and to analyze the underlying control problems. In fact, we perform a sort of reverse engineering of the past and present procedures in the healthcare system to see if e^3*control* can describe it. We separate the analysis in two e^3*control* cycles. In the first cycle, we perform a reverse engineering of the AWBZ system and, as a result, explain the design of the AWBZ system using the control theory in Chapter 2. In the

[1] In Dutch: Algemene Wet Bijzondere ziektekosten

second cycle, we analyze the weaknesses in the AWBZ system and suggest a future situation, which is not implemented in practice yet.

5.1 Research Context

The case study was conducted within the Freeband User eXperience (FRUX) Freeband project[2]. The FRUX Healthcare pilot explores opportunities for 'we-centric', context-sensitive bundles of service in the field of dementia care. One service on which the FRUX Healthcare pilot focuses is an intelligent, interactive, regional social chart that is easily accessible anywhere, anytime, and anyplace. Besides providing general information on care and welfare services to persons with dementia and their formal and informal caregivers, this Dynamic Interactive Social Chart for Dementia Care (DEM-DISC) also responds to individual needs by customizing care and support information and advice and by providing such relevant services as a planning and communication service. The social chart enables users to find relevant customized and context sensitive packages of care to match their needs, to resolve fragmentation issues and to improve accessibility of health care and welfare. For example, a carer of a person with dementia would find necessary services for the dementia patient in their region, such as day care, housekeeping assistance and meal delivery. Our role in the project was to model the governance mechanisms in the business model of the health care system and to investigate how the DEM-DISC will fit into the business model as a control service.

The data for the case study was collected by two researchers - the author of this thesis and J. Hulstijn - from a series of semi-structured interviews with five experts from different health care organizations. The interviewed were carried out with domain experts at the indication centre (in Dutch: centrum Indicatie Zorg (CIZ)), the administrative office (in Dutch: zorgkantoor) and the project partner Vrije Universiteit Medical Centre (VUMC). The resulting e^3*control* models were verified by the domain experts working for the indication centre and the VUMC. In addition, data was collected from publicly available documents [Okma, 2001, Exter et al., 2004] and web sites of the government[3]. The data were collected before 1 January 2006, when a new health care regulation came into operation in the Netherlands. Thus, some of the descriptions in this case study may deviate from the situation after 1 January 2006. The results of the case study were reported in various papers and reports including [Kartseva and Tan, 2005], [Kartseva et al., 2005c], [Kartseva et al., 2006c], [Kartseva et al., 2006d], and [Kartseva et al., 2007a].

5.2 Extraordinary health care system

To understand why the AWBZ care system should change in the first place, we perform an e^3*control* cycle for the situation as existed before April 2003. We will show that we are able to model several weaknesses of the system, about which we have learned from interviews with

[2] http://frux.freeband.nl

[3] www.zorgaanzet.nl, www.minvws.nl, www.pgb.nl, ww.cvz.nl, www.overheid.nl, all in Dutch.

experts and from the literature. We also model how these weaknesses have been resolved as the system has been changed in April 2003 and in January 2005. Although in reality the changes were motivated *not only* by the identified control problems, they still resulted in partial mitigation of the problems.

5.2.1 Step 1. Ideal value model

In Figure 5.1 we model the most important actors who participate in the AWBZ system. Every Dutch resident is entitled to receive AWBZ care. Therefore, we have a market segment of **residents**. The *OR-1* fork at the resident models that the residents either (1) pay taxes and do not use the AWBZ care, or (2) pay taxes *and* use the AWBZ care. In the last case, the path goes through the *AND-1* fork. In the last case, the resident also plays the role of a **patient**.

To enable the AWBZ funding for *some* of the residents, *all* residents have to pay taxes. This requires modelling of a value transfer *Taxes* from the residents to the government. The government is represented in the model by the Ministry of Health, Welfare and Sport, further referred as the **ministry**. In return, the residents get an insurance that the AWBZ services will be paid by the government, which is modelled with the object *AWBZ Insurance*.

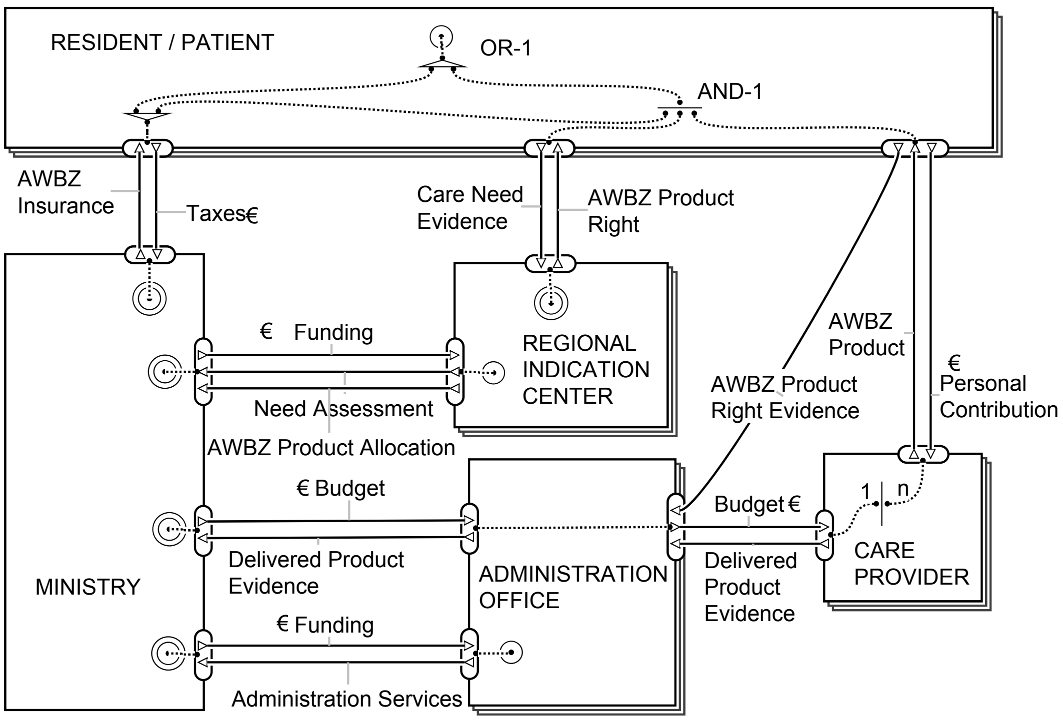


Figure 5.1: Value network for AWBZ health care system

The right path of the *AND-1* fork models receiving of *AWBZ Product* from the **care provider** by the resident in return for a *Personal Contribution*. *AWBZ Product* models a kind of treatment received by the resident. AWBZ law covers several types of treatments, e.g. help with cooking, cleaning, washing, etc. *Personal Contribution* is a small part of total costs of the treatment paid to care provider by the resident. The major part of the costs is covered by the government, and will be discussed later.

Not everyone should be able to use the AWBZ fund, but only those who have an illness covered by the AWBZ regulation. To avoid the misuse, in order to get access to the AWBZ product, a resident should prove that he or she has an illness that falls under AWBZ. The assessment of whether a person has such illness is done by an organization called the **regional indication centre**. The assessment is done based on evidence of the condition of the patient, like medical reports. In Figure 5.1 this evidence is modelled by a value object *Care Need Evidence*.

If a person qualifies for AWBZ care, the regional indication centre sends the person a *needs assessment letter*, which entitles this person to receive certain care covered by the AWBZ funding. Before April 2003 the need assessment letters specified what AWBZ products match patient's need. In fact, this letter gave the residents a *right* for certain AWBZ products. Thus, in Figure 5.1 we model a value object *AWBZ Product Right*, given to the residents in return for the *Care Need Evidence*. The *AND-1* fork that connects the transfers *AWBZ Product Right* and *AWBZ Product / Personal Contribution* models that the patient cannot get the AWBZ-related treatments for paying only the personal contribution without having the right for it.

When the regional indication centre grants a resident with the right by issuing the need assessment letter, a copy of this letter is send to an organization called **administrative care office** (in Dutch: Zorgkantoor). We model that this action corresponds to the value transfer of *AWBZ Product Right Evidence* to the administrative office. Note that the exchange of the evidence is between the resident and the administrative office, and not between the indication centre and the administrative office. This is because delivering of this evidence is a responsibility of the resident, not of the indication centre. In addition, the AWBZ product is granted in return for this evidence to the resident, not to the indication centre, as can be seen from the reciprocal exchanges of *AWBZ Product Right Evidence*, and *AWBZ Product*.

Administrative care offices, which operate on a regional level, make contracts with care providers. Only care providers contracted by an administrative care office can provide AWBZ services. In the Netherlands, the tasks of the administrative office are carried out by private insurance companies selected by the government. Normally, these are the biggest insurance companies in a certain region. Administrative care offices also operate on a region basis.

Administrative care offices also handle payments to care providers (see value object *Budget*). The care provider is paid on a budget basis. To provide accountability for the budget, the care provider needs to provide evidence of the delivered AWBZ care to the administrative care office. On the basis of such evidence, the accountability for the budget is established. This evidence is represented by a value object *Delivered Care Evidence*, and is given to the administrative care office by a care provider in return for *Budget*. Originally, the budget is given to the administrative care offices by the ministry. Administrative care offices are accountable for spending the budget. Therefore, they must provide to the ministry an evidence of the delivered care, to show how the budget is spent.

The introduced governmental agencies - regional indication centres and administrative care offices - are paid for their services by the ministry. Regional indication centres provide needs assessment services. In addition, because the need assessment letter contains information about specific products, the indication centres in fact do allocation of the patient's need to a specific

AWBZ product. Thus, we model that the indication centre gets from the ministry *Funding* and delivers *Need Assessment* and *AWBZ Product Allocation* in return. The services done by the administrative care office are modelled with a value object *Administration Services*, delivered to the ministry. In return the offices get *Funding*, similarly to the indication centre.

Note that we model the services (need assessment, product allocation and administration services) between the ministry and the agency, not between the agency and the resident. In fact, what we model is that the ministry *outsources* execution of the services to the agencies.

5.2.2 Step 2. Sub-ideal value model

As came out of interviews and literature research, there were several control problems in this system. To reduce complexity, we use two sub-ideal value models in Figures 5.2 and 5.3 to represent two different control problems. The models show two types of sub-ideal behaviour, which can both occur.

Control problem 1. Inadequate needs assessment

It is important to understand that regional indication centres are assigned to a specific region in the country. So, a resident can only apply to an indication centre in the region where he or she lives. In addition, the indication centre can only prescribe AWBZ products available in its region. In principle the entitlement to care should be the same for the whole country. But because different regions have a different care supply, there may be different ways of translating care needs into rights for care.

This situation replicates the situation before 1 January 2005. According to our experts, needs assessments turned out to be inadequate in some cases: patients from different regions were given different products for the same diagnosis. The decision of a regional indication centre was partly based on the supply of care in their region, and not on the health problems of the patient. This situation was not completely satisfactory, e.g. people from wealthier regions could get a better care than the people from less wealthy regions.

From a control perspective, the problem is a result of a lack of conflict of interest between the employees performing needs assessment and the employees performing the product allocation. The need assessment must be done objectively, whereas the AWBZ product allocation must be based on knowledge about the available products in the region. If these tasks are performed by one organization, the knowledge about products influences the decision about needs assessment.

To represent this problem, in the sub-ideal value model in Figure 5.2 we make the value transfer *Need Assessment* sub-ideal by marking it with a dashed line and naming the value object *Wrong Need Assessment*. In addition, because of the inadequate needs assessment a person can receive an improper medical treatment, e.g. due to a lack of facilities in a certain region. Thus, the transfer *AWBZ Product Right* is also sub-ideal as well as the *AWBZ Product*. In Figure 5.2 they are renamed to *Wrong AWBZ Product* and *Wrong AWBZ Product Right*. A liability token L1 is assigned to the regional indication centre to represent that this actor is responsible for causing the sub-ideal exchanges.

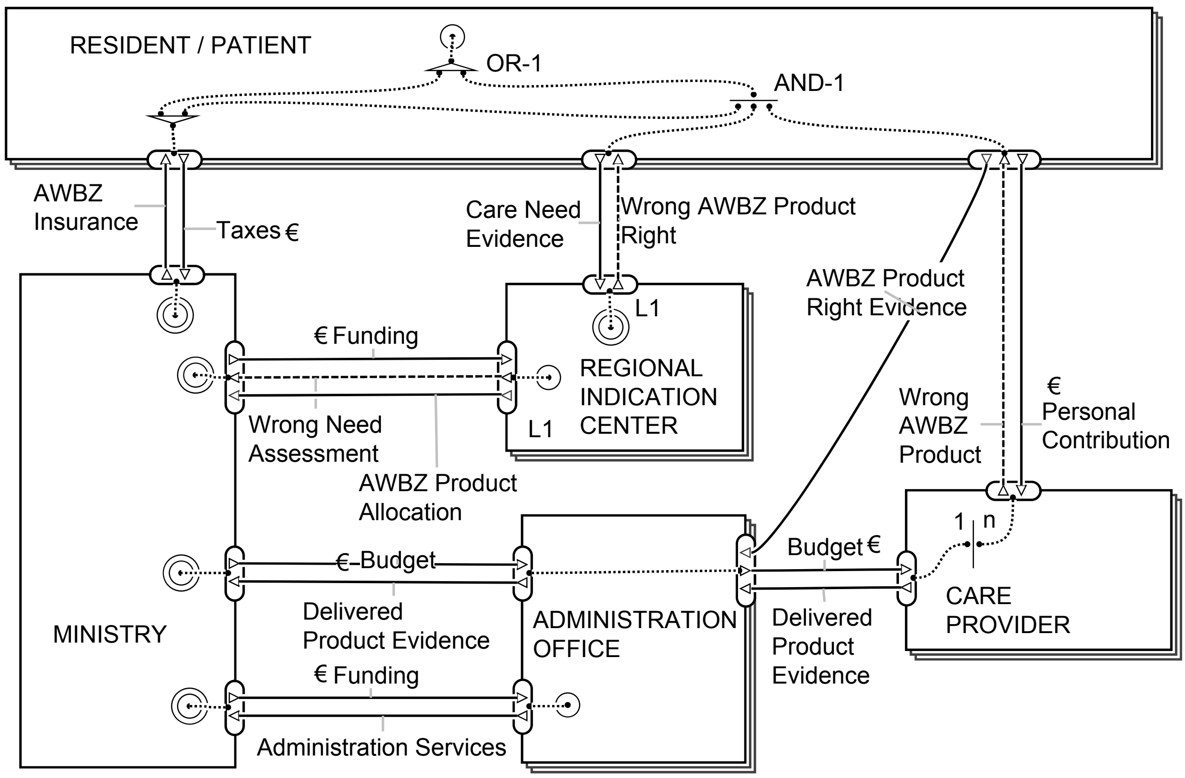


Figure 5.2: Control problem 1 in AWBZ: Inadequate needs assessment

Control problem 2. Risk of low quality care

In Figure 5.1, we do not have an actor who controls the quality of care products. As a result, provisioning of products of lower quality can remain undetected. To model this, in Figure 5.3 the transfer of *AWBZ Product* between the residents and care providers is marked as sub-ideal, and a liability token L2 is assigned to the care provider.

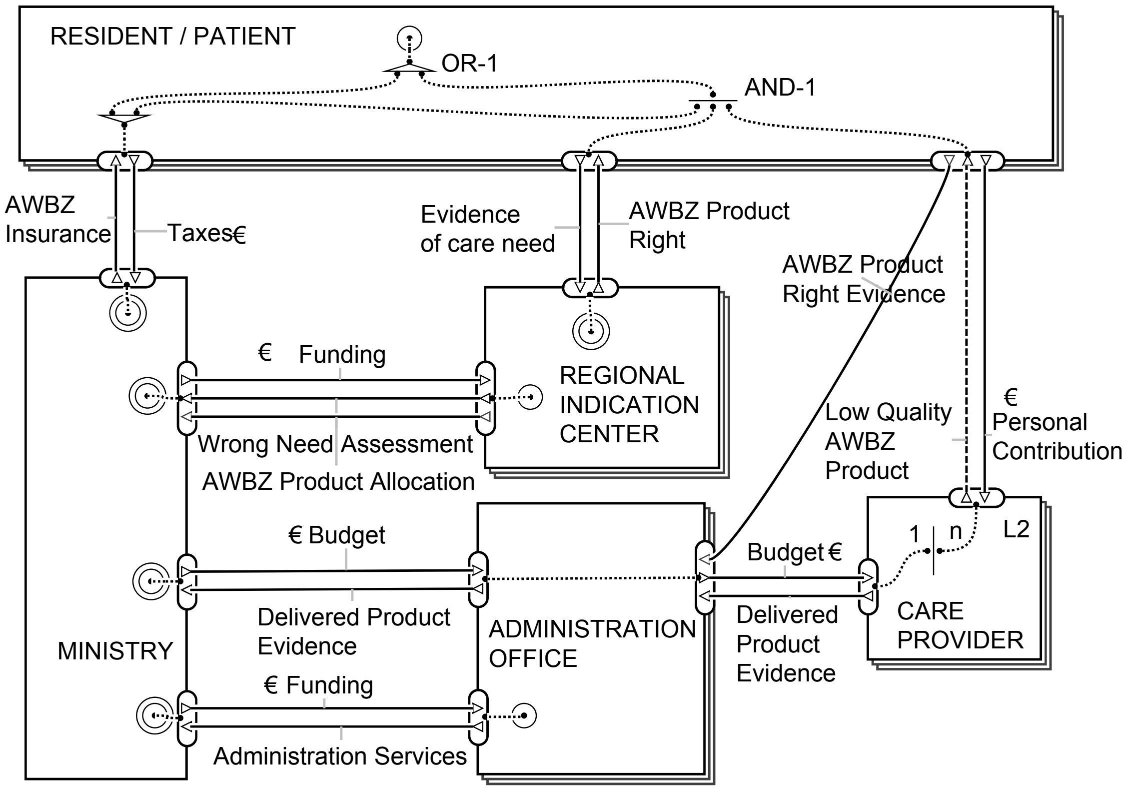


Figure 5.3: Control problem 2 in AWBZ: Risk of low quality care

5.2.3 Step 3. Control of indication centres and care providers

Control mechanism for problem 1. Splitting the regional indication centres

The government changed the system. In the new situation, the conflicting functions of needs assessment and product allocation are responsibility of two different agencies. As depicted in Figure 5.4, the needs assessment is done by *one indication centre*, a national organization created instead of the *regional* indication centres. Thus, instead of modelling a market segment of regional indication centres, we model an indication centre as an actor. The product allocation is now done by administrative care offices (see value object *AWBZ Product Allocation*). As a result, now there are two assessment steps.

In the first step, the need assessment is done by the indication centre. The needs assessment is formulated not in terms of products, as before, but in terms of *functions*. The AWBZ care is categorized in standardized functions, which include domestic care, personal care, nursing, supportive assistance, activating assistance, treatment and institutional care. For each function, the intensity of treatment is specified. Medical needs of a particular person are defined based on functions and intensities specified in the Care Entitlement Regulation[4]. As a result, the indication centre gives a resident a right to use certain functions (see *AWBZ Functions Right*), not to products, as previously. In addition, the indication centre now delivers to the ministry *Functions Need Assessment*, which stands for an assessment of a need for an function.

In the second step, the administrative care office 'translates' the right for AWBZ functions into a right for a specific product. In return for the right for product, a person has to demonstrate that he or she is entitled for AWBZ. This entitlement is nothing else than an evidence of the right for a function received from the indication centre (see value object *AWBZ Functions Right Evidence*). On the process level, this evidence is a copy of the needs assessment letter.

The changes were implemented incrementally. The functions were introduced on April 2003. The centralized indication centre was introduced in January 2005 [5].

Control mechanism for problem 2. Accreditation of Care Providers

To ensure that AWBZ care providers deliver quality services, an external controlling party is introduced. A basic form of quality control already does exist in the previous model. The administrative care office is in a position to select care providers who deliver AWBZ services. However, the administrative offices are mostly concerned with price negotiations rather than with quality controls. The ideal model does not contain a party who assesses the care provider.

Before being allowed to deliver care services, the ability of care providers to provide these services must be assessed. The result of this process is a kind of accreditation. The accreditation process is delegated to a specific semi-governmental agency: the Health Care Insurance Board (College voor Zorgverzekeringen (CvZ) in Dutch). The main task of the CvZ agency is to control health care insurance companies. Accreditation of care providers for AWBZ is one of their other functions[6]. This is modelled with an object *AWBZ Accreditation*, coming out of the CvZ actor.

In return, the CvZ requires the care provider to demonstrate their ability to deliver AWBZ services. This is modelled with the value object *Ability to Deliver AWBZ*. Thus, if a care provider does not have the ability to deliver the care, the accreditation is not granted.

In the present system in Figure 5.4, the administrative care office can only assign people to care providers who have an accreditation from the CvZ. Although the CvZ cannot provide an ex-post quality control, it can at least ensure that the institution has facilities to deliver adequate services (ex-ante control). An ex-post health care quality control is explained in section 5.3.

[4]Dutch: Besluit Zorgaanspraken AWBZ

[5]Source:Ministerie van VWS

[6]The responsibilities of CvZ were changed in 2006

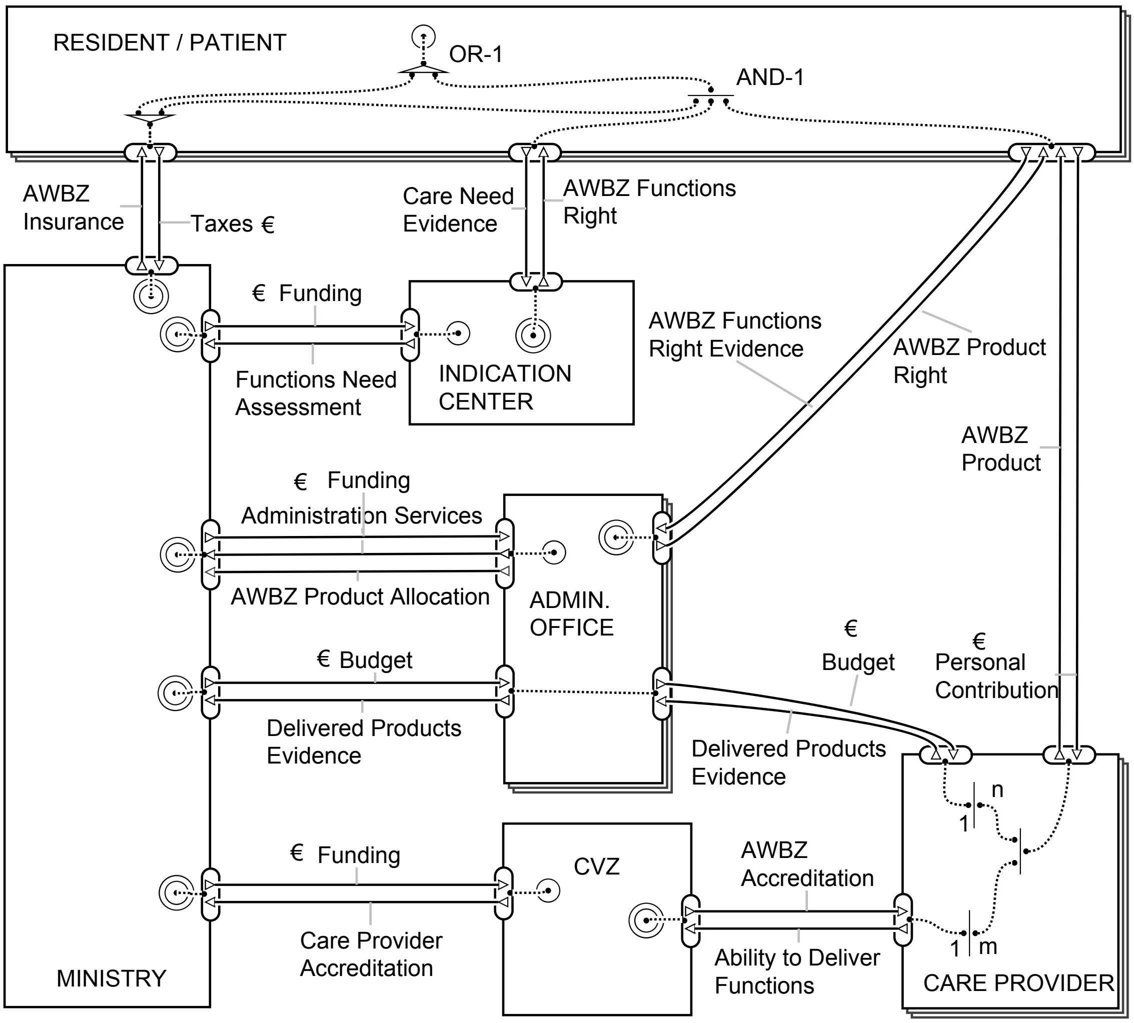


Figure 5.4: Solutions for the problems in the AWBZ health care system

5.3 Personal budget and Social Chart

The health care system described in the previous sections has still a number of problems. Decisions about issuing rights for a particular treatment, through needs assessment, are disconnected from the *available* treatments. Care providers cannot always deliver the prescribed AWBZ products, because their budget from the government has a limit. Because care providers do not have an incentive to provide services above their budget, the model in Figure 5.4 resulted in a *supply-driven system*, with queues of people waiting for a treatment. The task of the administrative care office is to manage those queues.

Before introducing the controls in section 5.2.3, the problem of queues was partly solved by regional indication centres, which had an overview of the available care in a region and were not prescribing the products that were not available. However, as we explained before, such solution was not really fair, since it the prescribed services differed from region to region.

In addition, older people, who represent the majority of AWBZ clients, are less willing to leave their home for a retirement house. The AWBZ services, contracted by the administrative care offices, mostly include services where people must live in a retirement houses or other house with medical facilities.

Nowadays, the Dutch government is moving towards a more demand-driven system. One of such changes is the introduction of a *personal budget* system. The new situation, the resulting control problems, and a possible solution are analyzed with the three steps of e^3*control*.

5.3.1 Step 1. Ideal value model

Firstly, in Figure 5.5 we present an ideal model of how the personal budget system works. Unlike in the previous system, with the personal budget a resident can select between being assigned to a care provider by the administrative care office (as previously) or selecting a care provider himself. In the last case, the administrative care office pays money not to the care provider directly, but to the resident. The resident selects the care provider and pays him. The resident can spend the money on any care services, provided either by traditional care providers, contracted by the administrative care offices, or by alternative care providers.

The alternative care provider may be any company. In some cases, where the AWBZ services do not require medical education, e.g. for cleaning, bathing, cooking, the alternative care provider can be even a family member or a friend.

Contracting alternative care providers is one of the objectives of the personal budget. It has a purpose to create more choices and to help disabled people to organize their own lives. For instance, older people do not have to wait for a place in a big care house, but can hire other people who support them in their live style at home. In addition, increasing the quantity of available care services should reduce the waiting lists.

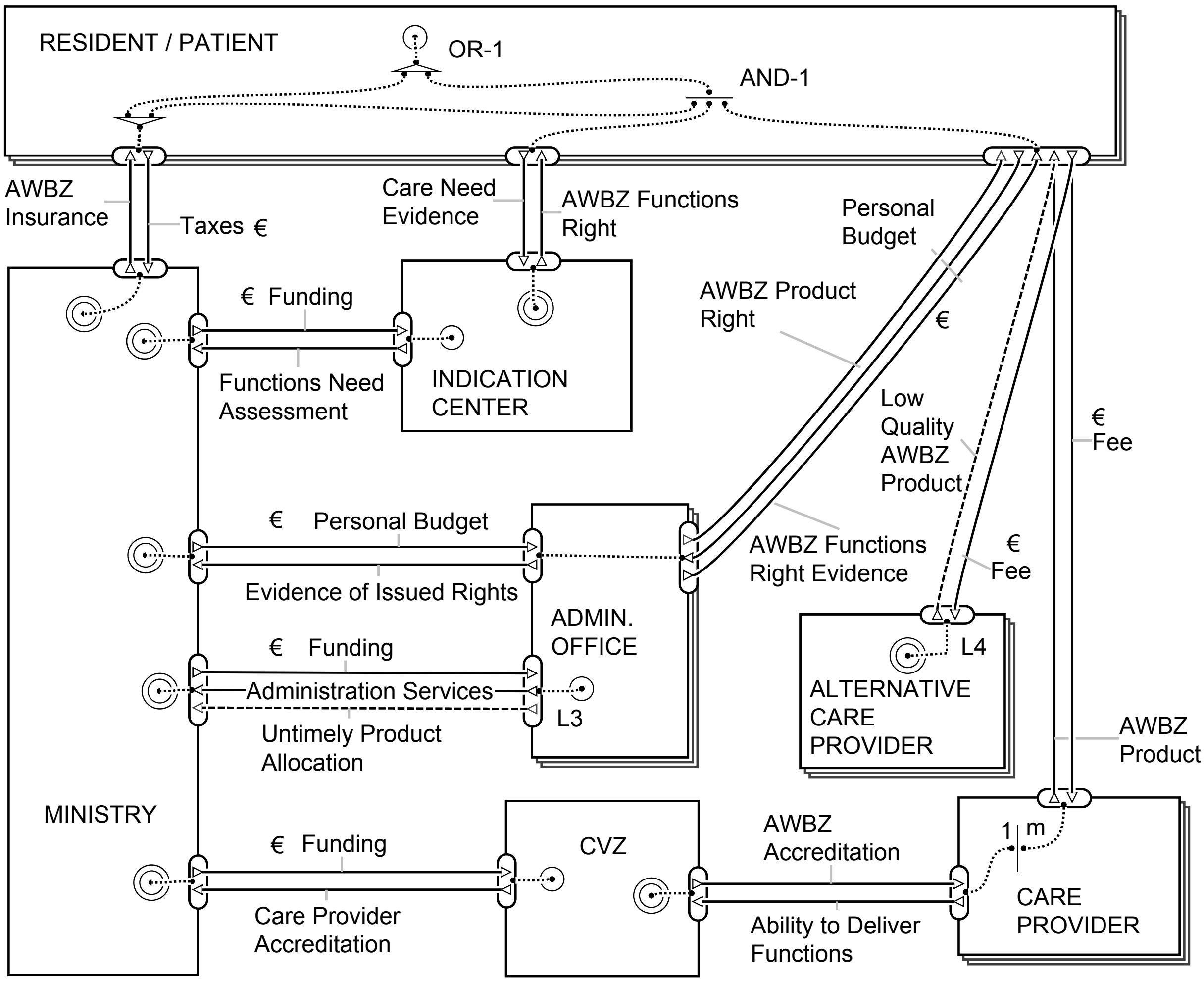


Figure 5.5: Control problems with the Personal Budget system

5.3.2 Step 2. Sub-ideal value model

The new personal budget system has some control problems. These problems are modelled in Figure 5.5.

Control problem 1: Information about alternative care providers.

As interviews revealed, currently people are not adequately informed about the available alternative care. The information about alternative care providers is not easy to find. Informing people about available care is a responsibility of the administrative care offices. An administrative care office has four tasks: (1) contracting care providers, (2) executing AWBZ, (3) product allocation and waiting lists management, and (4) informing people about available care and possibilities of AWBZ [7]. Information about care providers, available from the administrative care office, only concerns accredited care providers, with whom the offices make contracts. Administrative offices have no means to inform people about alternative care provider, since alternative care providers are not obliged to register at the administrative care offices. As a result, people tend to select traditional care providers, rather than alternative care providers, even if people use the personal budget.

This trend may stifle the development of the market for alternative care providers. This observation corresponds to the general idea that information asymmetry, a situation in which the customer has less information about a product than the provider, has a negative effect on the emergence of new markets [Bakos, 1998].

The personal budget was introduced to reduce the queues. However, if people do not use the personal budget as intended (namely to use alternative care providers), the queues will not be reduced. Therefore, in Figure 5.5 instead of the ideal value object *AWBZ Product Allocation* between the administrative care office and the ministry, this control problem is represented by a sub-ideal object *Untimely Product Allocation*.

Control problem 2: Quality of alternative care providers.

Traditional care providers have to be accredited by CvZ. Because of the large number of alternative care providers, the CvZ cannot accredit these similarly to traditional care providers. So, alternative providers are not required to have an accreditation. This results in a risk of low quality of services provided by alternative care providers. This is the same problem as control problem 2 considered in the first cycle in section 5.2. This problem is modelled by marking the value transfer of *AWBZ Product* between the resident and the alternative care provider with a dashed line and renaming it to *Low Quality AWBZ Product*.

5.3.3 Step 3. Controls: Social Chart

To solve the control problems 1 and 2 we suggest a possible future scenario, presented in Figure 5.6: a Social Chart. The Social Chart is an interactive web site that provides an overview of

[7]http://www.vgz-zorgkantoren.nl/

the care services in a region, and provides facilities for community-based quality control. The concept of such a dynamic interactive social chart, focused on the care takers of patients with dementia (DEM-DISC) is further developed in the Freeband User eXperience (FRUX) Freeband project [Droes et al., 2005].

Considering control problem 1: Care Provider Advice

In an e-commerce setting, the problem of information asymmetry would typically be solved by an information broker, who matches supply and demand. The Social Chart executes the role of such an information broker. The purpose of the Social Chart is to provide information about alternative care providers, and to help users to select a care provider that satisfies their individual needs. On the one hand, it provides facilities to give information about alternative care providers on-line. On the other hand, it provides facilities to search for a care provider on-line. In Figure 5.6 the Social Chart provides a service to residents called *Advice & Search*.

An interesting question is who will fund the Social Chart. In a possible scenario, the Social Chart would be funded by administrative care offices. An increased usage of alternative care providers as a result of the *Advice & Search* service offered by the Social Chart would reduce the waiting list problem and improve the information services, which are the administrative care offices responsibilities. Thus, the value transfer *AWBZ Product Allocation* becomes ideal.

Considering control problem 2: Informal Quality Control.

Quality control is a general concern in Dutch health care. Since 1995, an independent health inspectorate (in Dutch: Inspectie Gezondheidszorg; not in the model) supervises the quality of care providers. Given the expected explosion of new care providers, this organization cannot feasibly control the quality of all care providers. We, therefore, propose that the Social Chart should enable a kind of *informal quality control*. The Social Chart could provide, for example, a web-forum with testimonials, an online community peer review, a reputation mechanism, or collaborative filtering techniques [Schubert and Ginsburg, 2000]. In this manner, knowledge about the quality of care providers can be shared throughout the community of patients and relatives. Such community-based quality control only works when users contribute to the community. That is why in one possible scenario, depicted in Figure 5.6, the Social Chart receives a value object *Fill in Advice* from residents.

Since this informal assessment would reduce the administrative burden of assessing alternative care providers, one could argue that the ministry is interested to subsidize the Social Chart to stimulate the development of an effective virtual community. So, in Figure 5.6, a value object *Informal Quality Assessment* is exchanged from the Social Chart to the ministry in return for *Funding*.

Note that this is only one of the many possible scenarios. A social chart could be set up for example by the patients association, by commercial parties like an insurance company or information broker, or by local or central governmental bodies. Members of virtual communities in health care are generally willing to contribute to the community [Dannecker and Lechner, 2006]. We do realize that setting up a reliable system for online feedback is a research topic by itself.

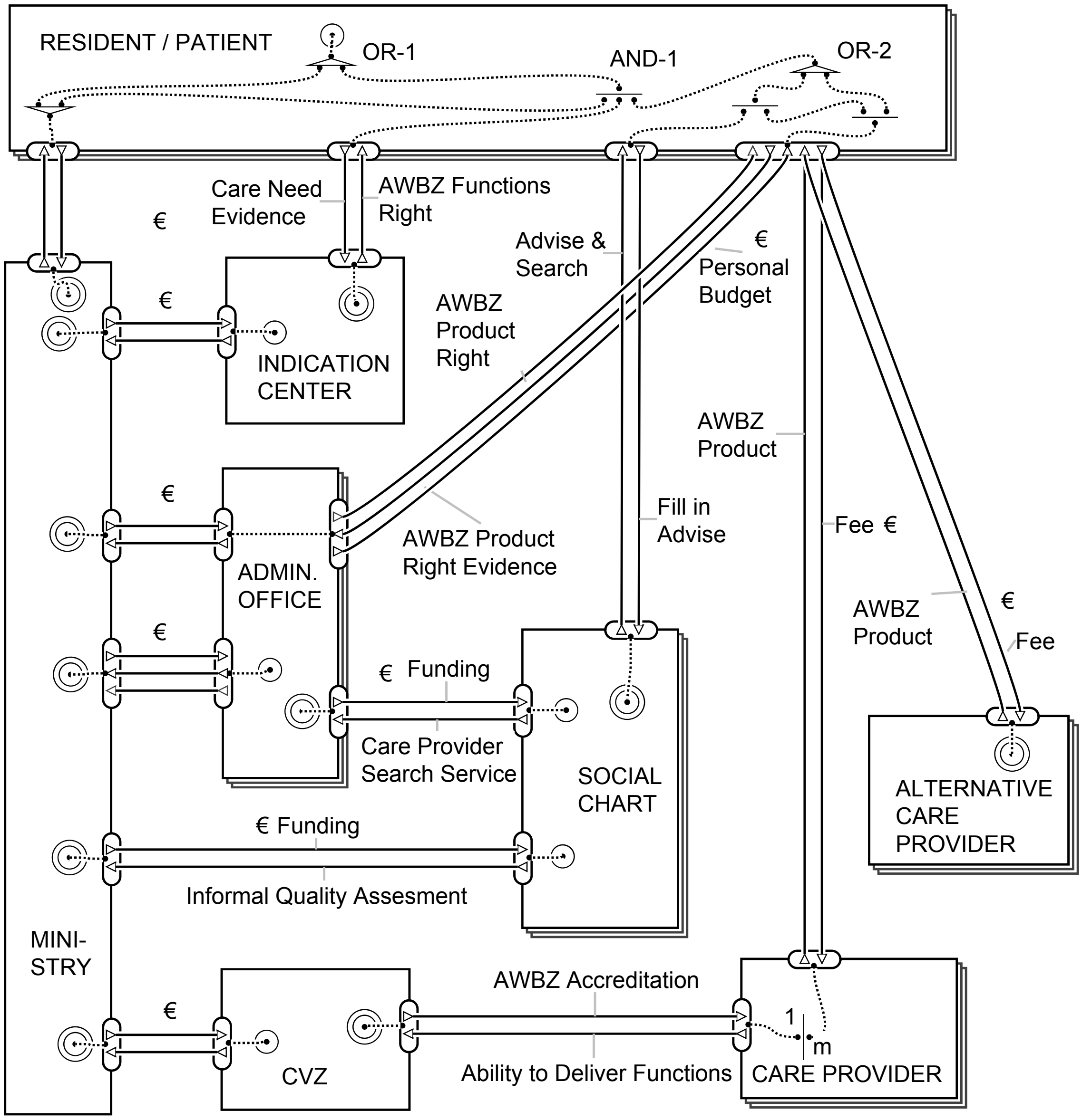


Figure 5.6: Social Chart as a solution

For example, testimonials tend to be biased. More quantitative comparisons also exist. For example a Dutch information broker, independer.nl, is using a large number of general practitioners to get statistically valid feedback on the health care services of various hospitals in the Netherlands.

5.4 Discussion

In this case study we have analyzed governance and control issues in the Dutch exceptional health care system (AWBZ), using the e^3*control* methodology, which is meant for the design and analysis of inter-organizational control mechanisms. The AWBZ case is interesting, because it shows the complexity of a highly regulated environment, which involves public-private partnerships. Since e^3*control* is based on an economic value perspective, it is reasonable to question whether such an approach is applicable in a sector like health care. In general, we argue based on this study that that e^3*control* is applicable also to design controls in a highly-regulated environment.

In addition, we have encountered some specific issues while developing this case study. In particular, they concern the design of indirect reciprocities, budgeting and community-based service. Below we discuss how we handle them in e^3*control* .

Indirect reciprocities. In the private sector, where the relations are often regulated by contracts, we mostly see direct reciprocities. A customer obtains a good or service and pays for it accordingly. But in the health care case described in this chapter, some economic reciprocal relations are indirect.

The provisioning of care is largely paid for by the tax payer. The system implements a *solidarity principle*, meaning that a majority of healthy people pays for an ill minority. In a way, residents pay for the guarantee that they will have access to health care of a certain quality, in case they would need it. In the value model we use an *OR-1* and *AND-1* forks to model that everyone pays taxes, while not everyone uses the care services.

Such solidarity system requires modelling indirect reciprocities. Without the solidarity system a reciprocity exists between a care provider and a resident: the care provider performs a medical treatment for the resident and gets paid by him in return. In the solidarity system in Figure 5.4 the money (see value object *Taxes*) and the treatment (see value object *AWBZ Product*) are modelled in different value interfaces and are exchanged even between different actors. For people who never suffer from a disease, covered by AWBZ, the reciprocity may never be realized. They pay taxes their whole life, but will never get a treatment, unless they have a need for it.

Modelling budgeting. In the private sector, we mostly see a direct relation between money coming to the actor and money leaving the actor. For example, if a wholesaler buys goods for some price, he usually sells it to a retailer for the same price plus a certain margin. In the health care case, we have many government agencies that are not paid per unit of their output. For example, the indication centre receives *Funding* from the ministry and issues an *AWBZ Right* to the residents. However, the indication centre does not receive proportionally more funding, when it issues more rights. In fact, it is paid for providing accurate, unbiased needs assessment. We model this with the reciprocal value transfers *Funding* and *Needs Assessment*. In addition, the value transfers of *Funding* and *AWBZ Function Right* are not connected by a dependency path, which shows that their valuation functions are not *directly* related. A similar situation is the case for other governmental agencies. At the administrative care office, the value transfers *Funding* and *AWBZ Product Right* are not connected with the dependency path; at the CvZ, the value transfers *Funding* and *AWBZ Accreditation* are also not connected.

Another type of indirect relations between incoming and outgoing value objects can be modelled with *cardinality jumps*. In the initial model in Figure 5.1 the care providers are paid based on a budget and not on per-treatment basis. Unlike the funding of the government agencies, the incoming value object *Budget* is still a function of the outgoing value object *AWBZ Product*. Thus, these two exchanges are connected by the dependency path. However, to be correct we model a cardinality jump: the care provider receives one budget and spends in on *n* treatments. With such a budgeting system, care providers cannot always deliver the AWBZ products to every patient who has the indication. Their budget is limited. Thus, if the budget is exhausted, the care provider either has to ask for a budget extension or stop delivering the AWBZ services. Getting budget extensions from the government is difficult. Because care providers do not have an incentive to provide services above their budget, the model in Figure 5.4 results in a *supply-driven system*

Community-based services. The Social Chart is a community-based control service. Such a community-based service, like a feedback or recommender system of the Social Chart, only works when sufficient members actually contribute to it. To represent such a community-based service in e^3*value*, requires reciprocity on the level of a market segment (stack of actors). This means that the users of the Social Chart benefits from the contributions of other users.

For the Social Chart, we model that the residents fill in an advice about a care provider (see *Fill in Advice*) and get in return a possibility to read this advice (see *Advise & Search*). Although the value objects are reciprocal, their reciprocity occurs not for one actor, but for one market segment: one resident fills in some advice, but it is read by another resident. The reciprocity does not occur at the level of an individual actor. In fact, some residents only read advices, but do not fill in anything in return. The reciprocity and, thus, functioning of the Social Chart, is possible when the whole market segment of the residents contributes to its functioning, and not one particular resident.

Such a reciprocity on the level of market segments is not explicitly defined in e^3*value*. Thus, to model the *community-based services*, some minor changes would be needed to the e^3*value* ontology, namely modelling the reciprocity on the level of market segments. This is not a control-related issue, so we do not consider it in this research.

5.5 Lessons learned

In the summary of lessons learned we reflect on our expectations about value modelling for controls and compare them to the results of this case study.

Lesson 1. Controls as commercial services with added value. We have seen that a highly regulated environment requires many control services. Examples in the case study include needs assessment (indication centre), allocation of actual care services, and the allocation of personal budgets (administrative care office), and providing information about care providers, and care provider assessment (Social Chart). Control services can be seen as commercial services, which can in principle be outsourced. This is clearly modelled in the e^3*control* models of this study: all controlling parties, whether they are government agencies or not, need to be funded or paid. Taken that the controls are seen as services, modelling them from the economic value perspective is very reveals another aspects of controls, as compared to the process models.

Lesson 2. Value aspects of control instruments. This case study illustrates numerous evidence documents. Examples of in Figure 5.4 include *Care Need Evidence*, *AWBZ Function Right Evidence*, and *Delivered Care Evidence*. This case confirms our arguments in favour of modelling evidence documents as value objects in certain contexts. Evidence documents are required to obtain other objects (rights, AWBZ care) and are thus of economic value for the actors. For example, a person cannot obtain AWBZ care without the needs assessment letter (modelled as the *AWBZ Product Right Evidence* value object).

Another reason why some evidence documents are of value for some actors is because they are needed for their *accountability*. Accountability is a core objective of many government agencies, as those in the health care case. Therefore, for them evidence is of value. For example, the indication centre needs the evidence for delivered care to be able to prove that the needs assessment is done properly.

Another control instrument with a value aspect is a *right*. A public-private partnership is often characterized by extensive regulations. Examples in the case study are *AWBZ Function Right* and *AWBZ Product Right*. Regulation can take the form of a system of rights, to restrict access to a service. Legal rights can be seen as value objects because they guarantee access to a product or service, and consequently, such rights are of real economic value to actors. For example, a right to an AWBZ product or a right to an AWBZ function is of value to residents, because it allows them to get access to AWBZ care.

In addition, the case confirms the argument of Chapter 3 that it is sometimes necessary to model the transfer of a right of a resource and transfer of a resource's custody in separate value transfers. In this case, the right to get a treatment *AWBZ Right* and the treatment itself *AWBZ Product* are modelled as different value objects.

Lesson 3. The value viewpoint is an abstract rationale for procedural aspects of controls. This case study demonstrates that modelling controls with value models is possible. Several control problems have been identified at the start of the case study. As a result of the analysis, we have been able to model all control problems using the concepts of e^3*control*.

The e^3*control* methodology allows focusing on pure value-related aspects of controls, which provides additional functionality as compared to the process perspective on controls. In particular, we found the economic value perspective important in design of a new controls service, the Social Chart. For the health care domain experts of the FRUX project [Droes et al., 2005], this was actually the most important result. Modelling the Social Chart from the economic value perspective spawns off such issues as who will fund the Social Chart, what service does it provide and who will operate the Social Chart. A further e^3*value* analysis enables answering these questions.

The issue on who will operate the Chart is important also from the control perspective. As the internal control theory states, the operator of the Social Chart should be independent from the care providers or insurance companies, who would like to advertise the service. The e^3*control* methodology offers tools to design networks by taking both the value and control aspects into account.

Lesson 4. The e^3*control* ontology. In general, all e^3*control* concepts have been useful in describing the control problems we have encountered. Such concepts as sub-ideal value object and sub-ideal path provide means to model control problems. As in the music case, in this case we can also model reconciliation controls with the value interface construct. For example, in Figure 5.4 the four-port value interface models that for every AWBZ product there should be a proper right for an AWBZ product. If we have ten AWBZ products transferred, but only nine rights, the e^3*value* software tool will generate an error message. This models that a care provider should not provide treatments to people who do not have the need assessment letters; otherwise they have to be charged a full price.

5.6 Summary

In this chapter we have described a case study on control mechanisms in a Dutch health care system. We have demonstrated two cycles of e^3*control*. In the first cycle we model control problems and solution implemented in past. Thus, we do not design the solution ourselves, but describe already existing solutions. Two problems have been modelled: (1) conflict of interest within the indication centres and (2) the lack of quality control of care providers. The related solutions are (1) splitting the functions of the indication centres and (2) introducing quality control by a government agency CvZ. In the second cycle we have described present problems in the health care system. We have found these problems during interviews with experts. These problems include (1) lack of information on available care and (2) lack of quality controls of alternative care providers. A solution for these problems is a Social Chart, an interactive web-portal for dementia patients and their informal carers.

In the case study we model all the control problems and mechanisms that were discovered during the data collection stage. This demonstrates that the value-based approaches like e^3*value* and e^3*control* are not only applicable in commercial settings, but also in the highly regulated sector. In particular, regulatory rights and evidence documents, two mechanisms that are often used by public institutions to regulate private businesses, can be seen as value objects. We have also demonstrated how *indirect reciprocities* can be modelled in e^3*value* and how to model principles such as *community-based services*, *solidarity* and *budgeting* in e^3*value* . Finally, we have seen that also in the public domain, many control services should actually be seen as *commercial services*, which should be adequately funded. The Social Chart is one such control service discussed here.

Chapter 6

Elicitation of control patterns

In the Chapters 3, 4 and 5 have suggested and demonstrated the e^3*control* approach to designing controls in network organizations. However, e^3*control*, as presented in previous chapters, does not as yet provide us with any specific control solutions. Neither does it contain explicit theory on various kinds of control problems and control mechanisms, which a designer requires. The design of inter-organizational controls would be facilitated if e^3*control* offered knowledge on control problems and control mechanisms in network organizations. Such knowledge could facilitate the selection of a control mechanism for a particular control problem. For instance, it would help the designer to have a library of existing common control problems and descriptions of proven solutions for these problems. To incorporate this knowledge into e^3*control*, we propose a library of *control patterns*.

An e^3***control*** **pattern** is a description of a generic and re-usable control mechanism for a recurring and well-known control problem. Control patterns are inspired by the use of design patterns in architecture [Alexander, 1979] and software engineering [Gamma et al., 1995]. The idea is to capture the 'best practices' in the design of buildings, software, and later also organizational design [Tapscott et al., 2000], [Weill and Vitale, 2001], [Rolland et al., 2000] for different applications. The e^3*control* patterns specifically focus on the domain of controls and in particular on control problems that occur in network organizations.

In Chapter 2 we have already introduced the concept of patterns and argued in favour of organizing design knowledge for inter-organizational controls as patterns. To summarize, the advantage of a patterns approach is as follows. On the one hand, patterns allow us to describe controls in the form of problem-solution pairs, which is also common in the control domain. That is to say, the problem can be represented using a sub-ideal model, and the solution turns the sub-ideal model into an ideal model. On the other hand, the patterns allow us to incorporate design knowledge for both process and value aspects of controls and also to capture not only a textual description of the problems and solutions, but also to represent the problem and solution using graphical conceptual models. Finally, we can also describe relationships between different patterns, which allows us to capture interactions between the patterns. Such iterations describe which control mechanisms substitute or complement each other.

To use control patterns in the e^3*control* design process, we have first to build a library of patterns. In other words: we have to *elicit* them. In this chapter we will do so by considering the literature review in Chapter 2 and undertaking a commonality-variability analysis of the groups of controls extracted from it. We regroup the extracted groups of controls into *unique problem-solution pairs*, which, later in Chapters 8 and 9 are encoded into control patterns.

6.1 Patterns elicitation method

Pattern development usually includes the *identification*, *collection* and *codification* of existing knowledge [Fowler, 1997, Coad, 1992]. A more advanced method for patterns elicitation, called PattCaR, has been developed to support elicitation of patterns for business processes (in clothing manufacturing companies) [Seruca and Loucopoulos, 2003] and focusses on organizational design.

To elicit patterns for the control domain, we use an adapted version of PattCaR. The initial set of PattCar core activities, from which our elicitation activities were derived, comprise the following: (1) Define domain and analyze context; (2) Define domain core business processes and vocabulary; (3) Describe sub-domains in terms of existing generic business processes; (4) Develop sub-domain enterprise models for examples of businesses; (5) Define patterns for the sub-domain; and (6) Organize and interrelate patterns.

As can be seen, the PattCaR method considers elicitation of patterns from large numbers of examples. In our case, the control domain is a well-established field of practice, which offers a wide range of documented examples that are synthesized in a theory, outlined in Chapter 2. Therefore, instead of analyzing the examples, we extract patterns based on the analysis of the control theory.

In practice, our elicitation process included the following **three phases**:

1. In the **first phase**, we became acquainted with the control domain and collected some examples of controls from case studies. This has been presented in Chapters 4 and 5.

2. In the **second phase**, we undertook a review of control problems and mechanisms described in the internal control theory, namely in [Romney and Steinbart, 2006], which lists multiple control problems and mechanisms associated with every activity of each transaction cycle. We reviewed the *Expenditure Cycle* and the *Revenue Cycle*, since they contain external activities which are of interest to us (see section 2.1 for more details on the cycles). Altogether, there are 12 control problems for the Revenue Cycle[1], and 17 control problems for Expenditure Cycle[2]; each control problem corresponds to about three or four control mechanisms. So, altogether we have about a hundred control mechanisms from [Romney and Steinbart, 2006], which also include both procedural measures and measures related to information systems. The latter were not considered.

   The control problems and mechanisms which were obtained are formulated in the original source [Romney and Steinbart, 2006] from the internal perspective, which is usual in the internal control literature. The control problems are defined under an implicit assumption

[1] See p. 381 of [Romney and Steinbart, 2006]
[2] See p. 435 of [Romney and Steinbart, 2006]

that a problem occurs either because an employee neglects a problem caused by an external party or because the employee himself causes the problem. Consequently, control mechanisms do not involve the activities of an external party.

Therefore, to obtain inter-organizational controls, we 'translated' the obtained internal controls to the network perspective, which implies that a control problem is caused by an external party. For example, for an internal problem 'Purchasing from unauthorized suppliers', the corresponding inter-organizational problem would be 'Supplier is unauthorized'. For the control problem 'Purchasing goods of inferior quality' the inter-organizational control problem would be 'Supplier sells goods of inferior quality'. The control mechanisms were interpreted in a similar way. After this 'translation' we grouped the control problems and mechanisms following several principles. Firstly, the controls have to form non-overlapping problem/solution pairs. Secondly, they have to be domain-independent: abstract away from the consumer/supplier relationship and expressed in terms of principal/agent relationship. We also abstracted such details as naming a specific document, e.g. a purchase order or a contract. Any document was considered to fall into the category 'statement'. As a result of this exercise, we have obtained domain-independent descriptions of controls, which has produced a list of *potential patterns*. These patterns have been reported in [Kartseva et al., 2006b].

3. In the **third phase**, we have discovered that the mechanisms described in the *potential patterns* are described in other sources. In fact, some mechanisms were the screening, signalling, monitoring and incentive mechanisms of the the agency theory (see section 2.1.2). Other mechanisms turned out to be the commitment evidence and execution evidence mechanisms described by Bons (see section 2.1.3). The *potential patterns* did not give a one-to-one match to the mechanisms described in literature. For example, we had one pattern *Partner Screening*, whereas the agency theory describes two groups of controls that could be described by our pattern - *Screening* and *Signalling*. Therefore, we have analyzed the commonalities and variabilities of the potential patterns and the mechanisms from literature. As a result, we have arrived at the set of final control patterns presented in this thesis and they are documented in Chapter 8 and Chapter 9.

In this section we only present the **third phase** of the elicitation, which is based on the agency theory, as described in section 2.1.2, and the work of Bons, as described in section 2.1.3. We will not present the second phase, since, as we have already explained, the patterns can be elicited solely from the control theories considered in the third phase. In addition, the internal control theory, which was used in the second phase, has already been described in detail in Chapter 2.

The patterns elicitation process of the third phase contains the steps listed below. In the field *'Source'* we state where in this thesis the step is described.

Step 1: Define domain and analyze context. In this step, we review the domain of controls and define its scope and context. This analysis is based on the thorough literature review undertaken in Chapter 2.

Source: Chapter 6, section 6.2

Step 2: Describe *sub-domains* and perform commonality-variability analysis. As already explained, the sub-domains are extracted from the control theories. We analyze each sub-domain with respect to its uniqueness when compared to the other sub-domains. They are compared and common and variable features are defined and then a list of control patterns is elicited.

Source: Chapter 6, section 6.3

Step 3: Define domain vocabulary. The domain vocabulary is a compendium of precise definitions of all significant terminology used by experts when discussing problems and solutions in the domain. The vocabulary is needed to make the patterns more precise and more understandable and it provides a terminology commonly understood by patterns users. Our vocabulary is based on the control theories, as reviewed in Chapter 2 as well as on a more in-depth review of the work on the formal modelling of controls of [Chen and Lee, 1992] and [Bons, 1997] in section 7.4.

In addition, we suggest a *pattern template*, which is a structure to describe each pattern, e.g. Context, Problem, Solution, Example, Variations.

Source: Chapter 7

Step 4: Define patterns. In this step, each candidate pattern is encoded according to the pattern template. In addition, the context, problem and solution of each pattern are described using value and process models, which are built using the concepts described in the patterns vocabulary and developed in the previous step.

Source: Chapter 8 and 9

Step 5: Organize and interrelate patterns. The relationship between the control patterns is stated in the 'Related patterns' part of each pattern. There we describe which other patterns provide alternative solutions.

In the remaining sections of this chapter we extend the review of literature to extract those control patterns, which correspond to Step 3 of our elicitation method.

6.2 Domain of inter-organizational controls

In Chapter 2 we have made an overview of the domain of controls in networks. The findings of Chapter 2 boil down to the following:

- Inter-organizational controls can be described in terms of the principal-agent framework. In such a framework, a distinction is made between a *primary* and a *counter actor*, where the primary actor plays the role of *principal*, and the counter actor plays the role of *agent* (see section 2.1.2). The counter actor behaves *sub-ideally* and the primary actor wants to reduce the loss caused by such behaviour.

- There are several types of control problems and mechanisms relevant for network organizations described in the agency theory (see Figure 2.1). These are Screening, Signalling, Monitoring, and Incentives. The Screening and Signalling mechanisms are used to counter the hidden characteristics problem. This occurs when the primary actor does not have enough knowledge about the counter actor's abilities or characteristics, as a result of which there is a risk that the counter actor will perform his activities in a sub-ideal way. The screening and signalling mechanisms recommend checking the counter actor's abilities and characteristics before signing a contract with him. The Monitoring control is used to counter the hidden action problem, which means that the counter actor performs his activities in a sub-ideal way. The monitoring mechanism recommends verifying the counter actor's performance before rewarding him. If monitoring is difficult or costly, the counter actor can be stimulated to behave ideally by Incentives which can be positive (reward) or negative (punishment).

- Two additional inter-organizational controls are described in [Bons, 1997]. The Commitment Evidence control applies to a situation in which a counter actor inappropriately denies his commitment to the primary actor. The Execution Evidence control applies to a situation in which a counter actor inappropriately claims that the primary actor executed his activities sub-ideally. Both commitment evidence and execution evidence controls require the creation of evidence that can be used in (legal) disputes against the counter actor. These two controls stem up from the audit trail principle of the internal control theory.

- A distinction can be made between *ex-ante* controls, i.e. controls executed before the contract between two actors is settled, and *ex-post* controls, i.e. controls executed after the contract is settled. The screening, signalling and settlement of incentives are ex-ante controls, while monitoring, commitment evidence, execution evidence and execution of incentives (actual rewarding or punishment) are ex-post controls. A further distinction can be made between *contractual* controls and *procedural* controls. Contractual controls employ value-based mechanisms to stimulate the counter actor to behave ideally. Procedural controls employ process-level mechanisms to repressively prevent or detect the counter actor's sub-ideal behaviour. With the exception of incentives, all the groups of controls considered here are procedural.

- It has already been argued that informal controls, which are important for network organizations, are more a condition rather than a mechanism that can be designed (see section 2.1.4). As in [Bons, 1997], the notion of trust can be incorporated in *condition* of inter-organizational controls, not as a mechanism. An inter-organizational control is only needed if there is not enough trust between primary and counter actors.

6.3 Commonality - variability analysis

In our review of the domain we have identified Screening, Signalling, Monitoring, Commitment Evidence, Execution Evidence and Incentives controls. We call these *sub-domains* of the control domain, but it should be noted that this is by no means an exhaustive list of sub-domains.

However, taken the focus and limitations of this work, we consider it to be a good starting point to elicit patterns. In this section we compare the sub-domains with each other, and re-group them to select the unique problem-solution pairs. These pairs will form the control patterns.

In order to extract the control patterns from the sub-domains, we perform the commonality-variability analysis which is used to identify the common features of similar models and the elements distinguishing them from other models. From this analysis, a set of control patterns will be elicited.

6.3.1 Grouping criteria

Before proceeding with the commonality-variability analysis, we must ascertain several requirements which we want our patterns to comply with. These requirements define the criteria we follow in identifying similar and variable features of the sub-domains. The following criteria are used:

- **Unique problem-solution pairs.** We want the patterns to represent a unique combination of problems and solutions. We establish the following criteria: (1) two different control problems should fall in two different patterns, (2) two different control mechanisms for one control problem should be represented in two different patterns, and (3) if one control mechanism mitigates two fundamentally different control problems, the two problems with the similar control mechanism should also be represented by two different patterns.

- **Lightweightness.** In line with the approach of *e^3value* [Gordijn, 2002], we want the patterns library to be lightweight. This means that the fewer patterns we have to describe all the considered controls, the better it is. To achieve this, we abstract from domain-specific details, which are present in the internal control theory. Firstly, we do not consider any specific roles of actors, such as a supplier or a customer. We describe a transaction in terms of the principal-agent framework, or, more exactly, its interpretation by [Bons, 1997], as described in section 2.1.3. In this way, we distinguish between a *primary* and a *counter actor*. The counter actor behaves *sub-ideally* and the primary actor wants to reduce the loss caused by this sub-ideal behaviour. The actors can delegate their activities to *trusted parties*. Secondly, we do not differentiate controls because they involve different types of documents, e.g. a purchase order or a contract. We only consider one object of type 'statement'.

- **Differences.** Although patterns are domain-independent, we do not want to make them too general. Therefore, we have defined some boundaries to distinguish certain differences between controls. To be specific, a difference is made between ex-post and ex-ante controls and contractual and procedural controls. For example, if two control problems or mechanisms rely on a similar process or a value transaction, but one mechanism is ex-ante and another is ex-post, we consider them to be two different mechanisms.

6.3.2 Dealing with delegation

The agency theory and the work of Bons, from which our patterns are extracted, consider controls in a relationship between two actors: a primary and a counter actor. In addition, Bons also considers some network aspects. He sees networks as being derived from a two-actor network as a result of the **delegation** of activities to other actors. But, should we include such situations with delegations in our patterns?

From practical reasons, this is not a good idea, since a very large number of different networks can be formed as a result of delegation. The corresponding activities of a primary and a counter actor can be delegated to each other or to a third party. Third parties can also further delegate these activities. If we also consider that actors not only execute primary and counter activities, but also other activities associated with controls (e.g. reconciling, witnessing, verifying), even more possibilities for delegation arise.

So, if we take delegation into account too, we will never be able to describe an exhaustive list of patterns and we will only end up with an enormous number of them. Furthermore, these patterns will describe similar control problems and mechanisms and only differ in the way activities are delegated, which, strictly speaking, is not a control issue.

On the other hand, the principles of controls in a delegation network are the same as in the two-actor transaction from which this network is derived. For example, a buyer always checks if a delivery is correct, and is not concerned about whether the delivery is made by the seller or if the seller delegates the delivery to a carrier. The only difference is that in the first case the buyer controls activities of the seller, and in the second case - of the carrier. Consequently, if we abstract from the specific actors difference and focus on the properties of the actors, the controls in the two situations are the same.

An important issue that has to be considered in delegation segregation of duties principle. For example, an actor should not have to control his own activities as a result of delegation. Therefore, the actors to whom an activity is delegated should have the properties that obey the segregation of duties. Such constraints can be defined also for the two-actor transaction between a *primary actor* and a *counter actor*, e.g. stating that an activity that controls a counter actor should not be delegated to this counter actor or his allies.

Taking this into account, we choose to describe each pattern only for a transaction between a primary and a counter actor. In order to describe delegation situations, we introduce **delegation patterns**, which provide guidelines on how the two-actor model of a control pattern should be properly changed into the multi-actor model. They ensure that when an activity is delegated, the controls prescribed by the control pattern, are still in place. Delegation patterns are described in Chapter 10.

6.3.3 Deriving the patterns from control sub-domains

In this section we perform the commonality-variability analysis of the control sub-domains. The following sub-domains have been identified: Screening, Signalling, Monitoring, Commitment Evidence, Execution Evidence and Incentives. We compare the sub-domains with each other to identify whether they represent one pattern or not. Where appropriate, we also look inside a sub-domain to consider whether it may be split up into several different patterns.

Screening and signalling

The controls of screening and signalling sub-domains mitigate the same control problem of hidden characteristics, see section 2.1.2. However, in the agency theory, screening and signalling are considered to be different control mechanisms. In screening, the primary actor verifies the *activities* of the counter actor. For example, when hiring a person, the employer puts him to a test and checks how the test is performed by actually observing the test activities or their results. In signalling, the primary actor verifies indirect *signals* and not the activities. Such signals have a historical correlation with the expected performance of the actor in the future. For example, an employer hires candidates who do well at school since it is known from the past that the chance is higher that they do the job better than the candidates who study not so well.

The difference between screening and signalling is that screening is based on information collected by direct observation of the counter actor's activity, while signalling is based on information collected from a third party. In the employment example, the indirect observation of the candidate's abilities is done by the university teachers that grade the candidate. However, if we ignore delegation, the difference between the two mechanisms disappears. In other words, signalling is the same as the screening, the only difference being that screening is performed by a third actor.

So, the screening and signalling describe the same control problem and, in our view, the same control mechanism. Therefore, they can be described by one pattern which we call **Partner Screening**, meaning that the primary actor screens his partner.

Screening and monitoring

Screening and monitoring mitigate the same control problem: a sub-ideal execution of contractual agreements by a counter actor. On the other hand, the screening control also considers a condition of hidden characteristics, and, therefore, employs an ex-ante control. As explained before, screening verifies activities performed by a counter actor in the past with the assumption that the past resembles the future. The monitoring mechanism does not carry the assumption of hidden characteristics. As a result, it is an ex-post control performed in the context of an existing contract and it suggests verification of activities under the contract.

Because of this difference, we describe screening and monitoring controls in different patterns. The pattern for monitoring control is called **Execution Monitoring**, meaning that the execution of the counter actor's activities is monitored by the counter actor.

Positive incentives and negative incentives

Incentives may be positive or negative. Positive incentives stimulate the counter actor to behave ideally by rewarding him while the negative incentives do the same by punishing the counter actor. As we have demonstrated in Chapter 9, these two mechanisms require different changes in value models. Namely, positive incentives can be created by adding an incoming value object to the counter actor in the case of ideal behaviour, while negative incentives can be created by adding an outgoing value object to the counter actor in the case of sub-ideal behaviour. We therefore put positive and negative incentives into two different patterns. Positive incentives are described in the **Incentive** pattern and negative incentives are described in the **Penalty** pattern.

Monitoring and incentives

Both the monitoring and the incentive mechanisms mitigate the same control problem - that of a sub-deal execution of contractual agreements by the counter actor. On the other hand, the two controls are different, the former is a procedural control, while the latter is a contractual control. Incentives also require monitoring mechanisms to check when the reward or punishment has to be issued or not. For example, fraudulent activities have to be proved before someone can be fined for tax fraud. Such proof can be modelled with the pattern Execution Monitoring, while the actions related to the punishment are a part of the Penalty pattern. So, as we have demonstrated in Chapter 9, the incentive mechanism is a *variation* of the monitoring mechanisms.

Execution evidence and commitment evidence

The execution and commitment evidence controls involve the same activity: the counter actor should provide the primary actor with an evidence document, which can later be used in a legal dispute. As a commitment evidence control, the evidence document contains a testimony of the counter actor's commitment to a future transaction with the primary actor. This evidence document is normally represented by a contract. For the execution evidence control, the evidence document contains the counter actor's testimony that the primary actor executed his obligations as stated in the contractual agreement. An example of such an evidence document is a receipt given as proof of payment.

So, the processes behind these mechanisms are technically the same, only the role of the evidence document is different. This is because the two controls address different control problems. In addition, the commitment evidence control is executed ex-ante, while the execution evidence control is executed ex-post. For these reasons, we describe these two controls in different patterns.

The pattern for the commitment evidence control is described in the **Proper Contracting** pattern. As the name implies, the control provides guidelines on a correct contracting process. The pattern for the execution evidence control is called **Execution Confirmation**. The name reflects the essence of the mechanism, which is to provide evidence about the execution of primary activities.

6.4 Control patterns

In the previous section we have identified several control patterns: Partner Screening, Proper Contracting, Execution Monitoring, Execution Confirmation, Incentive and Penalty. The patterns are summarized in Table 6.1, in which problems and their solutions are considered.

Table 6.1: Library of Control Patterns

Name	**Control Problem**	**Solution**
Partner Screening	Counter Actor executes his commitment sub-ideally.	Primary Actor verifies credentials of Counter Actor before making any commitments
Execution Monitoring	Counter Actor executes his commitment sub-ideally.	Primary Actor verifies Counter Actor's execution of the commitment, before executing own commitments
Incentive	Counter Actor executes his commitment sub-ideally.	Primary Actor provides a reward for the ideal execution
Penalty	Counter Actor executes his commitment sub-ideally.	Primary Actor provides a punishment for the sub-ideal execution
Proper Contracting	Counter Actor denies to have made a commitment to Primary Actor	Counter Actor provides an evidence document, which confirms his commitment
Execution Confirmation	Counter Actor denies that Primary Actor executes commitments ideally, and refuses to execute his commitments in return, or requires a compensation for executing his commitments	Counter Actor provides an evidence document, which confirms that Primary Actor executes his commitment ideally

Each pattern considers a transaction between two actors: a primary actor and a counter actor. The two actors make a commitment by signing a contractual agreement. A pattern describes a control problem and a control mechanism to mitigate this problem. In each pattern, the control problem is a result of the opportunistic behaviour of the counter actor, while the primary actor is the beneficiary of the control. The patterns are discussed in details in Chapter 8 and Chapter 9.

6.5 Classification of control patterns

As was explained earlier, the patterns Partner Screening, Execution Monitoring, Penalty and Incentive deal with the same control problem. What differs in these patterns are the context and control mechanisms. Partner Screening considers verification executed ex-ante, Execution Monitoring considers verification executed ex-post, and Penalty and Incentives consider contract-based punishments and rewards, agreed on ex-ante and executed ex-post. The knowledge about the ex-post or ex-ante nature of the control is included into the context of the pattern.

In general, a pattern is ex-ante if it describes a solution, executed *before* the contract is signed. A pattern is ex-post if it describes a solution, executed *after* the contract is signed. The Partner Screening and Proper Contracting patterns are ex-ante, while Execution Monitoring and Execution Confirmation are ex-post. The patterns Penalty and Incentive are executed both ex-ante and ex-post. Settlement of contracts where rewards or punishments are defined (e.g. agreements on ownership structure) take place ex-ante. However, the agents are rewarded or punished ex-post. This classification is shown in Figure 6.1.

As shown in Figure 6.1, the patterns can also be classified as contractual and procedural. Incentive and Penalty patterns involve contractual arrangements that settle financial incentives to align the interests of the parties. Therefore, these are contractual patterns. In fact, the contractual patterns provide value-based mechanisms to influence the behaviour of the counter actor so that he does not behave sub-ideally. The other patterns are procedural because they rely on procedures that prevent sub-ideal behaviour and do not involve financial incentives.

As with underlying controls, control patterns can be assigned to phases of a transaction cycle. The process of concluding a transaction consists of four phases: the preparation phase and the negotiation phase, which occur ex-ante, and the execution phase and the acceptance phase, which occur ex-post [Weigand and de Moor, 2003], see also [Bons, 1997], p. 30. Therefore, the Partner Screening pattern can be applied in the preparation phase and the Proper Contracting pattern in the negotiation phase. The Execution Monitoring pattern can be applied during the execution phase and the Execution Conformation pattern in the acceptance phase. The Incentive and Penalty patterns describe controls that are applied in both negotiation and execution phases.

6.6 Summary

In this section we have performed a domain analysis and commonality- variability analysis to extract six different *control patterns*. Our goal was to elicit a lightweight library of patterns, which represent unique problem-solution pairs.

Each pattern considers a transaction between two actors: a primary actor and a counter actor. A counter actor can behave sub-ideally, and a primary actor wishes to implement controls to mitigate the loss from such sub-ideal behaviour. Any company or person can fill the role of both primary and counter actor, depending on which control problem is considered.

Patterns are descriptions of control mechanisms for recurring control problems. Each pattern is a unique combination of a problem and a solution. A problem describes certain sub-ideal behaviour of the counter actor and the solution describes how the primary actor can mitigate the loss from such behaviour.

Patterns can be classified as *contractual* or *procedural*. Contractual patterns are Penalty and Incentive and procedural patterns are Partner Screening, Proper Contracting, Execution Monitoring and Execution Confirmation.

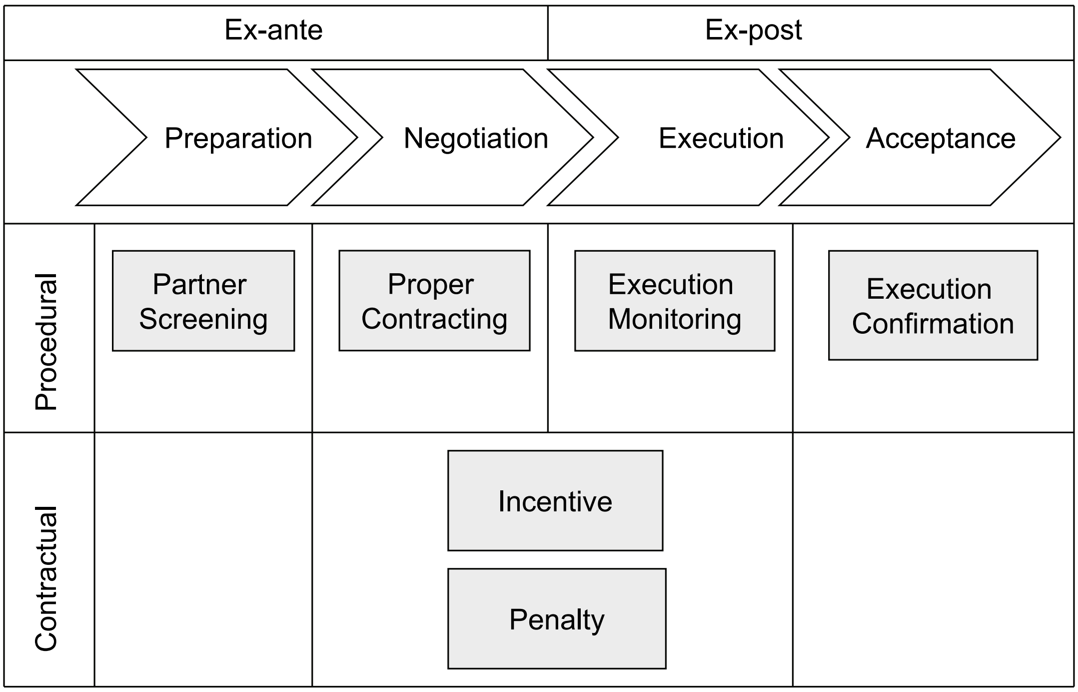


Figure 6.1: Control Patterns Classification

Another classification of patterns is *ex-ante* and *ex-post*. Knowledge about the ex-post or ex-ante nature of a control is included in the context of the pattern. Partner Screening and Proper Contracting are executed ex-ante. Execution Monitoring and Execution Confirmation are executed ex-post. Penalty and Incentive are special cases because they consider contract-based punishments and rewards, agreed on ex-ante, but executed ex-post.

In control patterns, activities which are needed to carry out a transaction, are executed by either a primary or a counter actor. To model situations, when a actors delegate some of their activities to third parties, we introduce *delegation patterns*. These patterns have rules on how a control pattern solution should be modified to model a situation with delegation, while ensuring that no control weaknesses occur because of delegation.

Chapter 7

Patterns vocabulary

In the previous chapter we put forward six e^3*control* patterns. These patterns have been described in an informal way using a textual description, see Table 6.1. Our goal is to represent these patterns using conceptual modelling techniques, which means that we want to describe the control problem and control mechanism of each pattern using conceptual models. We use e^3*value* and e^3*control* to model value viewpoint and some process modelling techniques for the process viewpoint. Conceptual modelling is necessary to come to a common graphical notation which will make the patterns more precise and achieve a common understanding between patterns users.

In real cases, the mechanisms suggested by the patterns are applied in complex environments and the pattern must be incorporated in other business processes and business models that are already in place. In such cases, we need guidelines to ensure that the patterns are implemented correctly. We need a mechanism to link the patterns with each other and with the external environment. To achieve this, in this section we develop a vocabulary of concepts, which we use to describe the control patterns. This vocabulary includes actors, value objects, operational activities and objects and is based on the control theories reviewed in Chapter 2 as well as on a more in-depth review of the work on formal modelling of controls of [Chen and Lee, 1992] and [Bons, 1997], which we undertake in this chapter.

We wish to describe the patterns using a minimum number of concepts in a vocabulary. To achieve this, we need to analyze each element which describes the patterns thoroughly, and develop a vocabulary with a minimum set of terms.

The vocabulary consists of three parts. The first part contains concepts needed to describe an *ideal situation*, in which no actor behaves opportunistically. The second part contains concepts needed to describe a *sub-ideal situation*, in which the counter actor behaves opportunistically. The third part contains concepts needed to describe control mechanisms and consists of two vocabularies: (1) a *Control Activities Vocabulary*, which describes types of control activities, used to models controls, and (2) a *Control Principles Vocabulary*, which describes how the activities should be embedded in the context so that control mechanisms are executed properly.

Before proceeding with the vocabulary we will give a more precise definition of a control pattern and describe a *pattern template*.

7.1 A control pattern template

Traditionally, a pattern consists of *name*, *context*, *problem*, *solutions* [Gamma et al., 1995]. We transfer the idea of using patterns to the control domain. We separate the description of the context in which a solution is to be applied from the problem which motivates the selection of a pattern [Coplien and Harrison, 2004].

In our interpretation, the context describes the value network with the actors, their relationships, including trust, and the activities to be controlled. We assume that no sub-ideal behaviour occurs. In other words, we describe an ideal situation in the context and we state at which moment of the transaction – ex-post or ex-ante – we will consider implementing the control. This knowledge is important to select a pattern. Some patterns consider the same problem and the choice between the two patterns should be based on additional information, such as that obtained at the moment of control execution.

The problem specifies sub-ideal behaviour by a counter actor which is to be mitigated by the solution of the pattern. The solution describes the value network and the corresponding process model, after implementing the control mechanism encoded in the pattern. We can therefore define the pattern in the following way. A *control pattern* is a description of a generic and re-usable control mechanism for a recurring control problem, selected on the basis of the context and it consists of the following elements:

name: a descriptive name of the pattern, used to select patterns from a pattern library.

context: a description of the business network to be controlled, modelled from an ideal perspective and which means that no one behaves opportunistically. The context also describes the trust relationship between the actors and the moment of solution execution: ex-ante or ex-post.

problem: a statement of opportunistic behaviour risks. A control problem exists if there is some deviation in the prescribed transfers of economic value.

solution: description of a control mechanism, to detect, prevent or correct the control problem.

We suggest the patterns template, shown in Table 7.1. Besides the textual description of the pattern, we also provide its graphical description which shows the context, problem and solution using conceptual modelling techniques. We concentrate on value and process viewpoints (see Chapter 2): To represent value aspects of the context, problem or solution, we use e^3*value* ; To represent process aspects, we use UML activity diagrams.

The context is interpreted in the control patterns as an ideal situation so it is represented by an ideal value model and a corresponding process model, also called ideal process model. The problem is interpreted as a sub-ideal situation and is therefore represented by a sub-ideal value model and a corresponding sub-ideal process model. Finally, the solution is a model of control mechanisms. It is represented by a value and process model with controls.

In the template we have also added a section Examples and optional sections Related Patterns and Variations. These items are also common for the design patterns [Gamma et al., 1995],

Table 7.1: Pattern Template

TEXTUAL DESCRIPTION	
Name	<name of the pattern>
Context	<description of the ideal situation>
Problem	<description of the sub-ideal situation>
Solution	<description of the control mechanism>
Example	<description of examples>
Variations	<description of variations of the control problem or mechanism>
Related Patterns	<description of related patterns>
GRAPHICAL DESCRIPTION	
Value View	**Process View**
<ideal value model>	<ideal process model>
<sub-ideal value model>	<sub-ideal process model>
<value model with controls>	<process model with controls>

[Coplien and Harrison, 2004]. Within Related Patterns we describe the patterns which offer alternative solutions. Within Variations we describe some of the most common variations of the patterns, e.g. some minor changes in process models that do not change the essence of the mechanism.

7.2 Ideal situation vocabulary

In this section we develop terminology to describe an ideal situation in the patterns used in the *Context* and *Solution* of the pattern. The solution is described with the use of ideal models because the controls, introduced by the solution, change a sub-ideal situation into an ideal one.

As was explained in Chapter 6, we describe a transaction between two actors in a pattern. We call them a *primary actor* and a *counter actor*. This terminology was inspired by [Bons, 1997], described earlier in section 2.1.3. In addition, we add the value perspective, absent in [Bons, 1997].

7.2.1 Value view

At the value level, the counter actor and the primary actor exchange value objects. The primary actor transfers a *primary value object PO* to the counter actor, and the counter actor transfers a *counter value object CO* in return. The graphical representation of this value transfer vocabulary is shown in Figure 7.1.

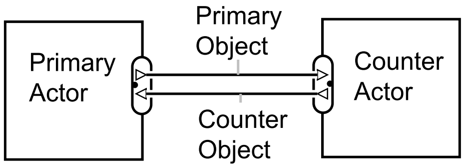


Figure 7.1: Control pattern vocabulary: ideal value view

7.2.2 Process view

An exchange of a single value object in a value model corresponds to the execution of one or more *operational* activities in a process model (see section 3.4 for the relationship between the value and process viewpoints). An operational activity, executed by the primary actor, which results in the transfer of the value object PO is called a *Primary* activity. An operational activity, executed by the counter actor, which results the transfer of the value object *CO* is called a *Counter* activity. These activities can also be collections of multiple operational activities, as defined in UML [Fowler and Scott, 2000]. In addition, one value transfer may require the execution of multiple operational activities.

The UML activity diagram in Figure 7.2 shows the execution of *Primary* and *Counter* activities by the primary and counter actors. This process model corresponds to the value model in Figure 7.1. The value model does not give us information about the order in which these activities are executed: The primary activity can be executed first, or the other way around. Unless additional information is provided by the context of the pattern, we assume that the order of activities is not relevant. To model this graphically in UML, we use the transaction fork (see Appendix B on the UML notation).

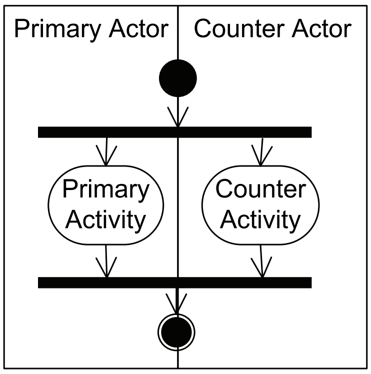


Figure 7.2: Control pattern vocabulary: ideal process view

Penalty and Incentive. A special kind of primary value object is the object *Incentive*. It is needed to model incentives given to the counter actor by the primary actor. At the process level, the transfer of the value object *Incentive* corresponds to the activity *Pay Incentive* which is a special case of primary activity.

A special kind of counter value object is the object *Penalty*. It is needed to model a penalty paid to the primary actor by the counter actor. At the process level, the transfer of the value object *Penalty* corresponds to the activity *Pay Penalty* by the counter actor which is a special case of counter activity.

7.3 Sub-ideal situation vocabulary

In the section *Problem* of a pattern we represent the sub-ideal behaviour of a counter actor. We model control problems only from the point of view of the primary actor which is within the framework of the agency theory (see section 2.1.2). The primary actor plays the role of principal and the counter actor plays the role of agent. As a result, by definition the counter actor behaves sub-ideally and the primary actor wishes to mitigate the risks caused by such behaviour.

This perspective does not mean that we only consider the counter actor to behave sub-ideally. Neither do we mean that the primary actor does not behave sub-ideally. So, each actor can play the role of both primary and counter actor. For instance, a pattern can be applied at least twice to a transaction between a buyer and a seller: (1) assuming the buyer is the primary actor and the seller is the counter actor, who behaves sub-ideally from the buyer's point of view and (2) vice versa, assuming the seller is the primary actor and the buyer is the counter actor, who behaves sub-ideally from the seller's point of view. Therefore, the intention is to handle the complexity of control problems in multi-actor networks by splitting them into parts of problems considered by every single actor.

In addition, we emphasize that we assume a *subjective view* on sub-ideal behaviour. As discussed in section 3.2.1, such behaviour may be perceived as sub-ideal by the actor, whereas another actor may perceive the same behaviour as ideal. Sub-ideality is defined by the primary actor in the patterns. For example, an untimely delivery of goods by a seller is sub-ideal from the point of view of a primary actor who is a buyer. A rational seller, interested only in maximizing his own wealth, might not consider the untimely delivery to be sub-ideal, unless it is settled in a legal contract. In some cases, a primary actor represents a group of actors, e.g. a certain market segment or an agency acting in the interests of a group.

7.3.1 Value view

To represent sub-ideal behaviour from the value viewpoint we use e^3*control* (see Chapter 3). The result of the counter actor's sub-ideal behaviour is a sub-ideal transfer of the value object *CO*, modelled as *Sub-ideal CO* in Figure 7.3. The counter actor has also been given a liability token *L* because he is responsible for this sub-ideal transfer.

7.3.2 Process view

In the process viewpoint, this sub-ideal value transfer *CO* corresponds to the execution of some operational activities of which we distinguish two types. The first type are ideal counter activities executed sub-ideally, i.e. not as prescribed in the ideal situation. We call them *Sub-ideal*

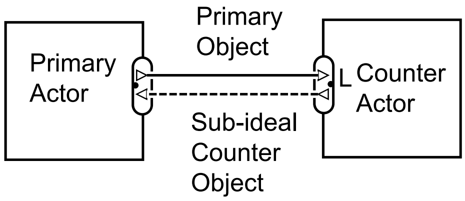


Figure 7.3: Control pattern vocabulary: sub-ideal value view

Counter activities. The second type, which we call *Emerging Sub-ideal* activities, are activities that do not exist in the ideal situation and that emerge as a result of the sub-ideal behaviour. We now consider the two types.

Sub-ideal Counter activities. A Sub-ideal Counter activity is an ideal counter activity not executed as prescribed by an ideal situation. Graphically the activity is modelled in Figure 7.4.

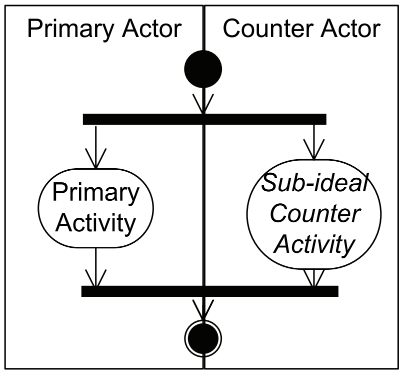


Figure 7.4: Control pattern vocabulary: sub-ideal process view

As described in Chapter 2, the following types of sub-ideal counter activities are possible:

- A counter activity takes place at the wrong time or in the wrong sequence (or does not take place at all)
- A counter activity is carried out without the proper authorization
- A counter activity which involves the wrong internal agent
- A counter activity which involves the wrong external agent
- A counter activity which involves the wrong resource
- A counter activity which involves the wrong amount or number of resources
- A counter activity carried out at the wrong location

In a situation when the value transfer of *CO* corresponds to more than one counter activity, at least one counter activity can be executed sub-ideally to cause sub-ideality of the value transfer *CO*. Furthermore, different combinations of sub-ideal counter activities can be related to different control problems, and therefore require different controls.

Emerging Sub-ideal activities. Sub-ideal behaviour may involve not only the sub-ideal execution of counter activities, but also the execution of additional activities that are not executed in a corresponding ideal situation. Such activities are called *Emerging Sub-ideal activities*. For example, activities are not executed in an ideal situation when a document is forged, e.g. printing the forged document.

Within this category of activities, we distinguish two special cases: *Deny Commitment* and *Deny Execution*. These two activities are necessary in order to model the sub-ideal behaviour in the controls described by [Bons, 1997], which are also a part of our pattern library, as described in Chapter 6. By executing the *Deny Commitment* activity the counter actor explicitly claims that he made no obligations to transfer the *CO* to the primary actor. By executing the *Deny Execution* activity the counter actor explicitly claims that the primary actor executed the primary activity sub-ideally.

7.4 Reviewing control terminology and principles

To derive a vocabulary for describing control mechanisms on the process level, we must return to the internal control theory and in particular to the control principles identified in section 2.1.1. The control principles - segregation of duties, authorization, access restriction, audit trail, and independent verification - describe procedural details on how a control should be implemented in a business process.

The plain text description of the control principles as given in section 2.1.1 is not very convenient from which to derive the vocabulary of controls design. To do this, we use the work on formal models of control principles by [Chen and Lee, 1992] and [Bons, 1997], [Bons et al., 1998], [Bons et al., 2000], reviewed in part in section 2.1.3, henceforth CLB. CLB focuses on the development of an expert system that automatically audits controls in business processes. As a result, they describe the control principles with more granulations and in a more structured way than the internal control theory does. In [Chen and Lee, 1992], the control principles are referred to as ‘auditing principles’. In [Bons, 1997] they are refers to them as ‘contextual conditions’. We provide a review of this work below.

Using the principles of CLB to develop the vocabulary is limited because that they do not cover all the internal control theory principles identified in section 2.1.1. They consider primarily the segregation of duties principle, independent verification principle and authorization principle. The access restriction, audit trail and adequate documents and records principles are not covered. However, in the scope of the current work this is not a restriction, since this latter group of principles is related mainly to the information system level of analysis.

7.4.1 Models of internal controls

To model control mechanisms, Chen and Lee have developed a set of what they call *auditing principles*, defined as "stereotypical descriptions of the relationships between activities, agents, assets, and information repositories involved in an internal control system[1]" [Chen and Lee, 1992]. The principles find their roots in the internal control theory as discussed in section 2.1.1, e.g. a segregation of duties principle. The original purpose of these principles was to use them as heuristics in an expert system. In such a system, control weaknesses in business processes are automatically detected by checking whether the business processes comply with the auditing principles. In Table 7.2 we present a summary of these auditing principles.

Table 7.2: Auditing principles of Chen

	Principles of precedence order of tasks
I	Whenever an *operating tasks* exists, a corresponding *control tasks* should also exist
II	Whenever an *operating task* and its corresponding *control task* exist, the *control task* should follow the *operating task*
	Principles of relation between information and tasks
III	When a *control task* exists, it must be furnished with *supporting documents*
IV	The *supporting documents* should be generated by a trustworthy source, e.g. by a previous *control task*
V	The *supporting documents* should be generated by a source *independent* of the source which generates the *document to be verified*
VI	The *supporting document* should be transferred directly from the *control task* that produced it
	Principles of organizational structure
VII	A *control task* and the *operating task* it intends to control should be segregated into two different *positions*
VIII	A *control task* and the *operating task* it intends to control should be delegated to two different *agents*
IX	The position responsible for a *control task* must not be lower in the formal power hierarchy than the position of the *operating task*
X	The agent responsible for a *control task* should be socially detached from the agent responsible for the *operating task*

The principles use a specific terminology. Firstly, they make a distinction between *operating tasks* and *control tasks*. A control task is a detective *verification activity*, which audits the results of an operating task with respect to its legitimacy, quality, or quantity [Chen and Lee, 1992]. A control task can therefore also be called 'verification task', and an operating task can also be called 'controlled task' or 'verified task'.

[1] Originally in [Chen and Lee, 1992] the principles are called 'control patterns'. However, their definition of the control patterns is defined from deontic logic perspective, and corresponds more to our definition of control principles, rather than to our definition of control patterns. In later work of [Bons et al., 1998] the patterns of Chen and Lee are referred as 'auditing principles'.

Furthermore, a control task compares a to-be-verified document and a supporting document. The *to-be-verified* document represents claims about an operating task (e.g. made by observing the task). The *supporting* documents represent trustworthy evidence about this operating task. For example, to verify whether the right goods have been delivered, we use the observation of the delivery and compare it with a purchase order or contract, which states what goods were ordered. In this case, the purchase order or contract is a supporting document and the observation of the delivery is a to-be-verified document.

The auditing principles in Table 7.2 are clustered in *three groups*. The first group considers the *order of tasks*. Within this group, Principle I states that every operating task should be verified by the control task. Principle II states that the control task has to occur after the operating task. The second group considers the *relationships between the tasks and the documents*. Principle III - V put additional requirements on the supporting documents, e.g. that they have to be present and that they have to be generated by an independent source. Principle VI requires the direct transfer of supporting documents from the source that generates them. The direct transfer requirement is crucial to avoid any tampering with the document. In practice, a very high percentage of fraud cases involve the alteration of otherwise valid documents [Chen and Lee, 1992]. Finally, the third group considers the *organizational structure*, which is, in fact, segregation of duties.

The operating tasks in Chen and Lee's model correspond to both the primary and the counter activity defined in our vocabulary. However, the patterns assume that the primary actor plays the role of principal and the counter actor the role of agent, and, they therefore, only take the sub-ideal behaviour of the counter actor into consideration. So, Chen and Lee's operating task is our counter activity and the Chen and Lee's control task is an activity that verifies the counter activity.

7.4.2 Models of inter-organizational controls

The auditing principles of Chen and Lee cover only internal controls of the expenditure cycle. Bons and others in [Bons, 1997], [Bons et al., 1998], [Bons et al., 2000] model controls in a similar way, but considers a broader set of them and specifically focuses on inter-organizational controls. As described earlier in section 2.1.3, Bons describes the controls that correspond to Execution Monitoring, Execution Confirmation and Proper Contracting patterns[2].

To describe the controls, Bons uses a set of rules, which are similar in structure to the auditing principles of Chen and Lee, but he calls them *contextual conditions*. They are listed in Table 7.3[3]. The set of contextual conditions is an addition to the set of Chen and Lee's auditing principles.

In addition to the *verification* activity (the control task in Chen and Lee), Bons distinguishes *witnessing*, *promising* and *testifying* activities. The witnessing activity is used in a strict sense: a party should either be present during the execution of the controlled activity, or it should be

[2]These controls are called General Principles in [Bons, 1997]

[3]In I [Bons, 1997] means that the document has to have a certain power. For instance, a bill of lading in international transport constitutes the right to take delivery of certain goods, whereas any other document does not constitute this right, although both documents may contain the same information.

Table 7.3: Contextual Conditions of Bons

	Contextual Conditions for Testifying and Witnessing
I	The right type of document should be issued by the proper role
II	The role that testifies should be different from the role that performs the primary activity
III	The parties playing the roles should be different and socially detached
IV	The role that testifies should witness the primary activity
V	This witnessing has to occur before the document is issued
VI	The document should be received by the role requesting the evidence
	Contextual Conditions for Promise
VII	The proper document should be issued by the appropriate role
VIII	The issuing role should be responsible for the primary activity being promised
IX	The beneficiary of this activity should receive the document

able to personally inspect some items through which the result of the activity is unambiguously determined. The latter is used in the definition of inspections, which do not usually take place during the actual activity that is being inspected. For instance, the production of goods is witnessed if a party actually sees the goods and is therefore able to infer that they have been produced.

Promising and testifying activities refer to the exchange of so-called 'performative' documents. Such a document is one of the instruments used as a part of a control mechanism and several functions of such documents can be specified using the Speech Act theory [Austin, 1962]. *Promising* refers to a promise that a party will perform some future action. *Testifying* refers to a statement made by a party that it has witnessed the completion of an activity.

7.4.3 Limitations of existing work

On the one hand, the work of CLB provides a good starting point to describe a vocabulary of control activities. The disadvantage of the Bons's contextual conditions for describing solutions to control patterns is that they are, as their name implies, *context-dependent*. That is to say, the conditions for testifying in Table 7.3 have been written for a situation in which a counter actor witnesses and testifies about a primary activity. However, they are inapplicable to other situations, e.g. where a primary actor witnesses and testifies about a counter activity. Similarly, the conditions for promising assume that a promise is made by a trusted party of the primary actor about the primary activity. The conditions for promising are not applicable to situations in which a promise is made about a counter activity by a counter actor. This limitation is also acknowledged by [Bons, 1997] at p. 60.

Chen and Lee's auditing principles are context-independent: every activity can play the role of an operational or control task. On the other hand, the disadvantage of this principles is that they only describe the principles for verification, and not the principles for witnessing, promising and testifying. In our case, we need all these principles to describe patterns.

Therefore, in our vocabulary, we take a joint set of control activities and control principles from both works. In addition, we express control principles as context-independent, which allows us to use them for each different pattern and not only for one specific pattern.

In general, our contribution differs from the work of CLB because we are aiming for a methodology to design new control mechanisms, instead of checking the correctness or trustworthiness of existing control procedures. In addition, unlike CLB, we do not intend to develop an expert system which uses automated reasoning to select the optimal controls for a particular network. Instead we have developed a methodology to assist human beings in the design process of controls. Our methodology is computer-supported in the sense that it supports the building of conceptual modelling diagrams for human designers.

7.4.4 The control activities and control principles vocabulary

In the next two sections, we suggest a vocabulary to describe control mechanisms in patterns. Firstly, we describe a vocabulary of control activities in which we define them. We describe their *required inputs and outputs* which are *obligatory*, meaning that it is impossible to execute the activity without satisfying the input/output conditions defined in the activity vocabulary. For example, it is impossible to execute verification of a document without first having seen it.

Secondly, in the *Control Principles Vocabulary* we specify *controls principles*, derived from the auditing principles of [Chen and Lee, 1992] and contextual conditions of [Bons, 1997]. The control principles are defined outside the context of a particular pattern. For example, in principle we describe a relationship between a testifying activity and a testified activity. In such a way, the role of the testified activity can be executed by any activity in the pattern: a primary activity, a counter activity or any other activity. We are therefore able to describe not only testification of a primary activity, as in [Bons, 1997], but also of a counter activity.

Finally, not all the auditing principles of [Chen and Lee, 1992] in Table 7.2 are used in control principles. The auditing principles were originally developed for internal controls. The inter-organizational dimension of this research brings with it an extra complexity and to compensate for it we simplify things by not taking some control principles any further into consideration. In particular, Principles VII and VIII distinguish between positions and agents within organizations. Positions are the responsibility of one particular person, so in this work we only model complete enterprises, and do not consider positions. Principle IX has also been omitted. At this stage of our research we do not define hierarchical relationships between enterprises. These are simplifications that can be added in further research.

7.5 Control activities vocabulary

In this and the following sections we describe concepts needed to model control mechanisms from the process perspective only. As we will see later, it is sufficient to use the ideal situation vocabulary from section 7.2 to model controls from the value perspective.

In this section we focus on operational activities needed to execute control mechanisms which are called *control activities*. *A set of control activities includes any activity needed for a control* *mechanism*. Note that our definition of control activities is broader than that of verification as in [Chen and Lee, 1992]. We describe the control activities *verify*, *witness*, *testify* and *promise*.

Graphical notation. Activities are represented graphically using the notation of UML activity diagrams. Take Figure 7.5 as an example. It represents an *actor* by using *swim-lines*, the two parallel lines at the sides of the figure. The actor executes an *activity*, which is represented by the *rounded square*, see *Verify Activity*. The inputs and outputs of the activity are modelled as *objects*, represented with by squares, see e.g. *Positive Statement*. Further details about the UML notation can be found in the Appendix B.

7.5.1 Verify activity

The *Verify* activity is defined as the verification of correctness, legitimacy, or completeness of some operating activity or its outcome[4]. The required inputs and outputs of the verify activity are presented in Figure 7.5.

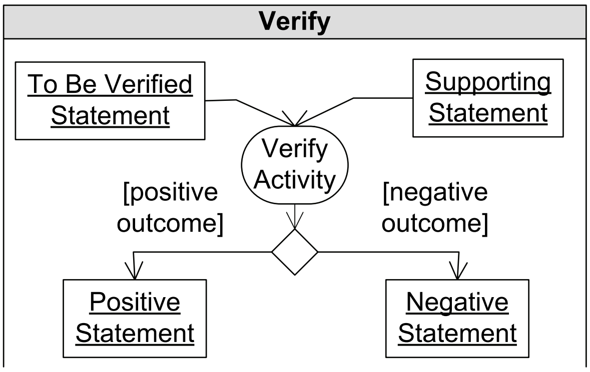


Figure 7.5: Activity 'Verify'

Statements. As shown in Figure 7.5, the *Verify* activity has two required inputs: a *To-be-verified Statement* and a *Supporting Statement*. We prefer to use the term 'statement' rather than 'document'. Nowadays, many paper documents have been replaced by electronic records. The term 'statement' is more general and can be applied to both paper documents and electronic records. We define 'statement' as an object, denoting either an oral or written statement, whereby the latter is an evidence document or an electronic record that has the status of such a document. In a control setting, the statement must be written, so that it can further fulfil its function as an evidence document.

To-be-verified and Supporting Statements. The *Verify* activity compares the information in the to-be-verified statement and supporting statements. The to-be-verified statement represents *untrustworthy* information that must be verified by the activity *Verify*. The supporting statement

[4]Our definition of the verification activity corresponds to the 'control task' in [Chen and Lee, 1992]

represents information that helps to judge the trustworthiness of the information in the to-be-verified statement. For example, if a seller claims that goods have been delivered, but the buyer has not seen the actual delivery, then verification lies in the fact of delivery. In this case, the to-be-verified statement is the seller's claim and the supporting statement is the buyer's observation of the actual delivery or *observed delivery*.

The roles of to-be-verified and supporting statements should be assigned depending on the *subject of verification* and thus one document can play both roles in different situations. In the previous example, the buyer wants to verify the *fact of the delivery*. In another situation, when a buyer has already observed a delivery and wants to check its *correctness*, the supporting statement is a purchase order or a contract between the buyer and the seller, which states what was ordered and what should therefore be delivered. *Observed delivery* is now the to-be-verified statement, whereas in the previous situation it played the role of the supporting statement.

Negative and Positive Statements. The outcome of the *Verify* activity is a *decision element* with two *guard conditions*: *positive outcome* and *negative outcome*. The guard condition 'positive outcome' is met if the *Verify* activity finds a correspondence between the to-be-verified statement and the supporting statement. Otherwise, the guard condition 'negative outcome' is met. If the positive outcome is met, the control flow ends with a *positive statement*; if the negative outcome is met, the control flow follows ends with a *negative statement*.

For example, if the verification of a delivery reveals that the goods were not ordered, the outcome of the verification activity is a document stating that the delivery is illegitimate. This is a negative statement. Otherwise, the outcome is a document stating that the delivery is legitimate. This is a positive statement.

We often omit the positive and negative statements in our models: the outcome can already be seen from the guard conditions *[positive outcome]* and *[negative outcome]* of a decision element. However, it is important to remember that the outcome of the decision element produces objects of type *Statement*. This allows us to connect the outcome of the verify activity to other activities that require objects of type *Statement* as an input, e.g. a testify activity.

7.5.2 Witness activity

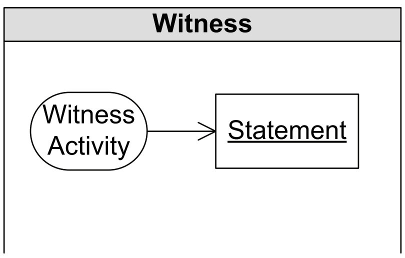


Figure 7.6: Activity 'Witness'

The *Witness* activity reports on an observation of reality[5]. In Figure 7.6 the output of the witness activity is an object of type *Statement* which contains information about the observation of reality. The statement produced by the witness activity can be further used for various purposes, such as an incoming object for a verification activity.

7.5.3 Testify activity

Unlike witnessing, the *Testify* activity testifies information in another statement, not an observation[6]. The difference between witnessing and testifying is that in the case of witnessing someone 'sees something happening and 'declares' that he has really seen it. In the case of testifying, someone 'declares' that information in a statement is true, an example of which is signing a document.

In Figure 7.7, the *Testify* activity requires an input of an object of type *Statement*, and has a *Testifying Statement* as an output. In addition, we consider a situation in which one actor testifies something to another actor. Thus, a *Testifying Statement* is transferred between two actors. This case is the most common in patterns.

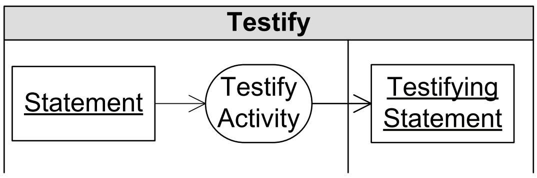


Figure 7.7: Activity 'Testify'

7.5.4 Promise activity

The *Promise* activity gives confirmation of some future actions. By giving a promise, an actor obliges himself to execute an activity in *future*. The promise activity in Figure 7.8 does not have any obligatory inputs: Anyone can promise anything without any previous requirements. The output of the promise activity is an object of type *Statement*, called *Confirmation Statement*, which is an evidence document about the promise made. An example of a confirmation statement is a contract in which a seller promises to provide some product or service to a buyer. In patterns we only consider promises made by a primary actor to a counter actor, or vice versa. Therefore, in Figure 7.8 we model a situation in which a confirmation statement is transferred between two actors.

[5]Note that our interpretation of witnessing is slightly different to that of [Bons, 1997]. Bons defines witnessing as merely observing something and not reporting about it. In Bons, the reporting is done by the testifying activity

[6]Note that we interpret testifying slightly different than that of [Bons, 1997]. Bons considers is as testifying about both observations and documents.

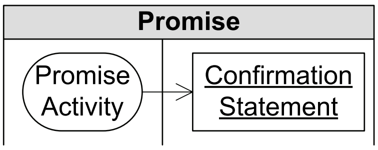


Figure 7.8: Activity 'Promise'

7.6 Control principles vocabulary

In this section we define the *control principles* for every control activity, described in section 7.5. The principles have their root in the internal control theory and, more specifically, are based on the auditing principles of [Chen and Lee, 1992] and contextual conditions of Bons, described in section 7.4. As such, we have control principles for verification, witnessing, promising, and testification. In addition, we introduce a pre-execution principle based on work of Bons and described in section 2.1.3. The difference between our control principles and the work of CBL is discussed in detail in section 7.4.3.

Structure of the control principles. We describe all the principles in a unified manner within three groups:

1. The *Activities-Activities* principles describe relationships between two activities, e.g. which activity should be executed first.
2. The *Activities-Objects* principles describe the relationships between an object. e.g. a statement and the activities that exchange this object. For example, these principles require a *direct transfer* of a statement between two activities.
3. The *Activities-Actors* principles describe which actors have to execute the activity and involves the *segregation of duties* principles.

The control principles vocabulary is an extension of the control activities vocabulary and it introduces more restriction on execution of a control activity. Unlike in the control activities vocabulary, in control principles vocabulary the necessity of these restrictions is defined by the control theory, not by the activity itself. For example, the control activities vocabulary states that the *Testify* activity must have a statement as an input, however it does not state that the statement should come from an activity executed earlier; the latter requirement is stated in the control principles vocabulary which follows on.

It is therefore physically possible to execute a control activity without the inputs and outputs specified in the control principles vocabulary. However, such an execution would be *untrustworthy* according to the internal control theory. For example, it will be untrustworthy to execute *Testify* activity before an activity which is being testified is executed. Similarly, it will be untrustworthy to execute *Verify* activity before an activity which is being verified is executed. Such restrictions are described in the control principles vocabulary. The control principles must hold in any control pattern and violation of a control principle creates a *control weakness*.

Graphical notation. Each principle is represented with a graphical notation in UML. Take Figure 7.9 as an example. There we see two variations of one principle: *A* and *B*. In variation *A* the principles require two activities to be executed parallel to each other. In variation *B* the activities are executed in a sequential order. This is modelled using the corresponding concepts of the UML activity diagrams notation: a control flow and a transaction fork.

The activities are assigned to different actors, represented by swim-lines (see the explanation of the UML swim-lines in Appendix B). We do not give any specific names to the actors. However, the principles are defined assuming the interest of the actor who executes the control activity. In other words, the actor who executes the control activity is a primary actor or his trusted party.

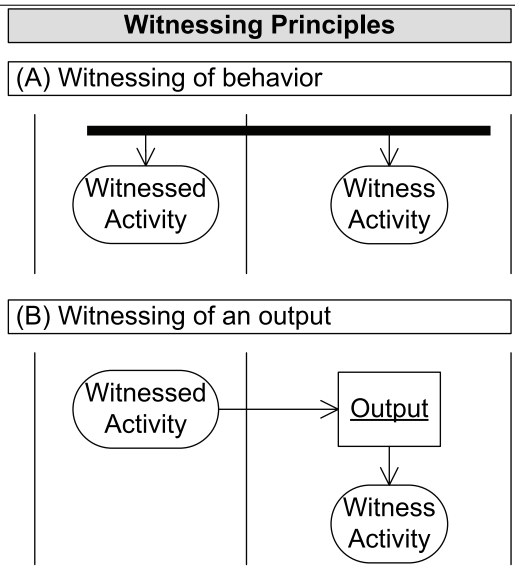
Segregation of duties in graphical notation. The swim-lines have a special semantics here. We use them to represent the segregation of duties requirement, written in Activity-Actor principles. For example, in Figure 7.9 we assign two activities to different actors, separated by a swim-line, to model that an activity *Witness* is executed by an actor who is not the one who executes the activity *Witnessed*. If activities are not separated by a swim-line, the model shows that it is not obligatory to separate them. This notation is used throughout the book to represent the segregation of two activities.

Numbering the principles. The textual formulation of the control principles is listed along with their graphical notation, see Figure 7.9. In this and other figures, the principles are given an ID, which consists of a letter (see *W*) and a number (I, II, III...). The letter refers to the name of the principle, e.g. *W* stands for the *W*itnessing principle. The letters *a* and *b* indicate to what variation of the principle we refer to. The Witnessing principle in Figure 7.9 has two variations: *A* for witnessing of behaviour and *A* witnessing of outcome. If neither *a* nor *b* are indicated, then the principle should hold for both situations.

7.6.1 Witnessing Principles

Every witness activity can be associated with an activity that is being witnessed and is referred to as the *witnessed* activity. Any operating or control activity can play the role of a witnessed activity, for example, that of witnessing a counter or primary activity, or even witnessing another witness activity.

We distinguish *two kinds of witnessing*: witnessing of *behaviour* and witnessing of *output*. This distinction corresponds to the behaviour and output control in the agency theory (see section 2.1.2) and in the management control [Ouchi, 1979]. Bons also makes this distinction implicitly by stating that the witnessing activity implies that (1) the party should either be present during the execution of the controlled activity, or (2) it should be able to personally inspect some items through which the result of the activity is unambiguously determined. The first part of the definition refers to the witnessing of behaviour, and the second part refers to the witnessing of output.



	Activities-Activities principles
W- I	Witness activity must exist
W-IIa	Witnessed activity must occur at the same time as Witness activity
W-IIb	Witnessed activity must precede Witness activity
	Activities-Objects principles
W-IIIb	Witnessed Outcome must be directly transferred to Witness activity from the Witnessed activity
	Activities-Actors principles
W-IV	An actor executing Witness activity must be independent and socially detached from the actor executing Witnessed activity.

Figure 7.9: Witnessing Principles

Witnessing of behaviour. For witnessing of behaviour, the observer wants to make a statement about the *execution* of the *Witnessed* activity. Thus, the *Witness* activity must occur *at the same time* as the *Witnessed* activity. In Figure 7.9 (A) we show this by modelling a parallel execution of the *Witnessed* and *Witness* activities. We cannot model the simultaneous execution in the UML language, and, therefore, we model the parallel execution only.

Witnessing of output. For the witnessing of output, the observer wants to make a statement about an *output* of an activity. The output is modelled as an object *Output* and it is assumed that the output is an object through which the result of the *Witnessed* activity is unambiguously determined. In this case simultaneous execution is not required. As shown in Figure 7.9 (B), the *Witnessed* activity happens before the *Witness* activity.

We often omit the object *Output* in our graphical models in order to simplify them. However, it is important to remember that non-simultaneous execution can only be allowed if the output of the activity *Witnessed* determines the witnessed activity *unambiguously*. For example, *Output* must not be altered by unauthorized people so that it represents trustworthy information about the *Witnessed* activity. If *Output* does not possess such characteristics, *Witnessed* and *Witness* have to be executed simultaneously, as in the witnessing of behaviour in Figure 7.9 (A).

The control principles for relationships between the activities *Witness* and *Witnessed* are formulated in text in Figure 7.9. The first principle requires the *Witnessed* activity to precede the *Witness* activity, the motivation behind it being that we cannot witness something that has not happened. The second principle requires the *Witness* activity to be executed by a party that is independent and socially detached from the actor who executes the *Witnessed* activity. This goes back to the segregation of duties: making claims about your own activities is untrustworthy.

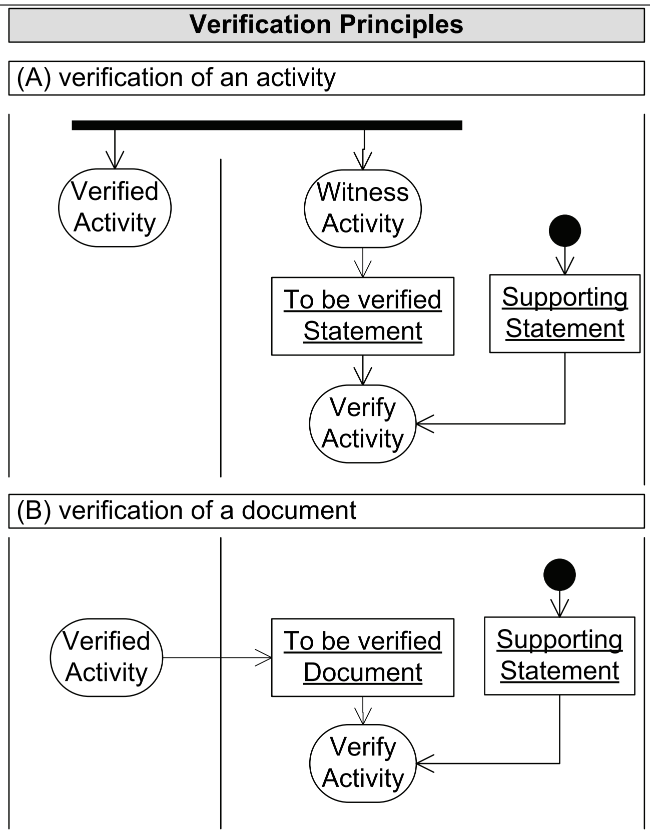
7.6.2 Verification Principles

Each verification activity corresponds to some activity which it verifies and will be further referred to as a *Verified* activity. In fact, any activity can play the role of a *Verified* activity. For example, we can speak about verifying a counter or a primary activity, verifying a witnessing activity, or even verifying a verification activity.

The principles for relationships between a *Verify* and a *Verified* activity are formulated graphically and in text in Figure 7.10. The *Verify* activity checks the correctness, completeness or legitimacy of the *Verified* activity. For example, if the *Verified* activity corresponds to a delivery of goods, then the *Verify* activity checks if the delivered goods were indeed ordered.

Verification of a behaviour. In section (A) of Figure 7.10 we considered the verification of behaviour or, in other words, verification of the execution of a *Verified* activity which should be observed by a *Witness* activity and only then verified.

The *Witness* activity is executed as required by the Witnessing control principle: by a party independent of the party executing the observed *Verified* activity and after the *Verified* activity. The *Witness* activity produces a *To-be-verified Statement*. The *Supporting Statement* is information that helps to judge correctness, completeness or legitimacy of the *Verified* activity,



	Activities-Activities principles
V-Ia	Witness activity must exist
V-II	Verify activity must exist
V-IIIa	Witness activity must follow Verified activity
V-IVa	Verify activity must follow Witness activity
	Activities-Objects principles
V-V	Supporting Statement must be directly transferred to Verify activity from a source that generates it
V-VIa	To-be-verified Statement must be directly transferred to from the Witness activity to the Verify activity
V-VIb	To-be-verified Statement must be directly transferred to from the Verified activity to the Verify activity
	Activities-Actors principles
V-VIIa	An actor executing Witness activity must be independent and socially detached from the actor executing Verified activity
V-VIII	An actor executing Verify activity must be independent and socially detached from the actor executing Verified activity
V-VIX	Supporting Statement must generated by an actor independent and socially detached from the actor executing Verified activity

Figure 7.10: Verification Principles

as described in the *To-be-verified Statement*. For more details on the definition of the *Verify* activity see section 7.5.1.

Verification of an output. In section (B) of Figure 7.10 we consider a *Verified* activity which produces a statement, for example, when the *Verified* activity produces a contract and verification is needed to check if the contract satisfies the necessary requirements. In such cases, the *Witness* activity, which refers to the witnessing of this statement, is not modelled: The UML activity diagrams assume that the receiving party possesses the statement, which is equivalent to witnessing the document.

The control principles for the *Verify* activity in Figure 7.10 are based on Chen's auditing principles described in section 7.4. The *Verified* activity must occur before the *Verify* activity and the *Verify* activity must be executed after the *Witness* activity. This is because it is not possible to verify something that has not been witnessed in the same way that it is not possible to witness something that has not happened. Furthermore, the *Verify* and *Witness* activities must be executed by an actor, who is independent and socially detached from the actor executing the *Verified*. In addition, the *Supporting Statement* must be produced by a source, independent and socially detached from the actor executing the *Verified* activity. This is related to the segregation of duties: witnessing, verifying or making some claims about your own actions is not trustworthy. In addition, in order to avoid forging, the *Supporting Statement* must be transferred directly from the issuing actor to the receiving actor. The same goes for the *To-be-verified Statement* in situation *A*.

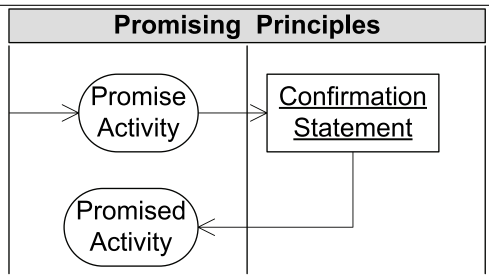
Generally speaking, the principle about the direct transfer of a statement can be relaxed if the statement cannot be forged, i.e. tamper-proof. Proving whether a document is tamper-proof or not is an information-level decision and it is out of the scope of this research. Therefore, in all patterns we assume that statements are not tamper-proof and we require direct transfer.

Henceforth, the source that generates the supporting statement is modelled as the *UML Initial State* element. Semantically, the UML initial state is a sub-class of the operational activity. Since we do not always know which activity generates the supporting statement, we model it with the initial state. According to the control principle V-VIX in Figure 7.10, the initial state should be assigned to an actor who is independent and socially detached from the actor executing the *Verified* activity.

7.6.3 Promising Principles

A *Promise* activity creates an obligation to execute certain activities in the future. These future activities are called *Promised* activities and a promise made by a primary actor creates his obligations with respect to the *Primary* activity. So, the *Promised* activity is the *Primary* activity for a primary actor and in a similar way the *Promised* activity is the *Counter* activity for a counter actor.

The control theory states that an actor should only execute activities if he has promised to do so. This is stated in the contextual condition VIII of [Bons, 1997] in Table 7.3. An actor should first sign a contract and only then should he execute activities under it. Therefore, in Figure 7.11 the *Promise* activity should always precede the *Promised* activity of the same actor. In addition, the direct transfer of a *Confirmation Statement* should be observed.



	Activities-Activities principles
P-I	Promise activity must exist
P-II	Promise activity must precede Promised activity of the same actor
	Activities-Objects principles
P-III	Confirmation statement must be directly transferred from the Promise activity to the actor receiving the promise

Figure 7.11: Promising Principles

7.6.4 Testification Principles

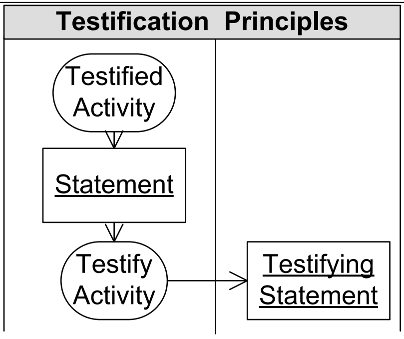
As defined in section 7.5.3, *Testify* activity testifies information in another statement. This statement is produced by a *Testified* activity, which we model in the principle here. The control theory prohibits testifying about anything that has not been executed. It is similar to the principle for the verification activity, which prohibits verification of any activities that have not been executed. Thus, the *Testified* activity that produces the statement must be executed before the *Testify* activity.

The principles for relationships between the *Testify* and *Testified* activities are formulated graphically and in text in Figure 7.12. It states that the *Testify* activity must follow the *Testified* activity. In addition, the *Statement* issued by the *Testified* activity must be transferred directly to the *Testify* activity. The same rule of direct transfer is applied to the *Testifying Statement*. The last principle ensures that only the actor who possesses the *Statement* can testify it.

When the *Testified* activity is represented by a *Witness* activity, the Testification Principles model the contextual conditions of Bons for witnessing and testifying activities as listed in Table 7.3.

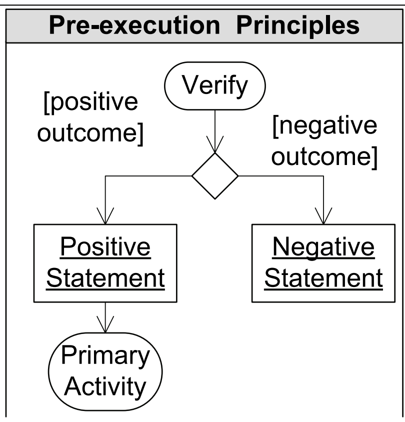
7.6.5 Pre-execution Principles

In addition to the control principles, which have been derived from the auditing principles of [Chen and Lee, 1992] and contextual conditions of [Bons, 1997], some additional rules are



	Activities-Activities principles
T-I	Testify activity must exist
T-II	Testify activity must follow Testified activity.
	Activities-Objects principles
T-III	Statement must be directly transferred from the Testified activity to the Testify activity
T-IV	Testifying Statement must be directly transferred from the Testify activity to the actor receiving the testification
T-V	Testify activity must be executed by the same actor who has the Statement

Figure 7.12: Testification Principles



	Activities-Activities principles
PE-I	Verify activity must exist (optional)
PE-II	Verify activity must follow Primary activity
	Activities-Objects principles
PE-III	Positive Statement must be directly transferred from the Verify activity to the Primary activity

Figure 7.13: Pre-execution Principles

needed to describe patterns. These rules are the **pre-execution requirements**, described earlier in section 2.1.3. The pre-execution requirements are found in Bons, who *implicitly* states them within the definitions of the controls, see [Bons, 1997], pp.60-62. For instance, in monitoring controls, the pre-execution requirement is to execute the verification of the counter activity before the primary activity is executed.

An important feature that makes the pre-execution requirements different from the other control principles is that they can be relaxed if the *trust* relationship between the primary and the counter actor changes. For example, in monitoring controls, the verification of a delivery may be done after payment is made if the buyer trusts the seller. ‘Trust’ means here that the buyer expects the seller to refund the goods or to exchange them if any damages is discovered. If the buyer does not trust the seller, he will check the goods before paying, as required by the pre-execution principle.

The pre-execution principles for the verification and primary activities are listed in Figure 7.13. As indicated, the existence of a *Verify* activity is optional.

The pre-execution principle does not necessarily only apply to the *Verify* activity and the *Primary* activity and the patterns contain different variations of the pre-execution principle. They include the requirement to execute verification before promising and to execute a promising activity only after the counter actor’s promising activity. All the variations are not described here and are only described within the patterns. The common feature of all these variations in different patterns is that they can be relaxed depending on the level of trust between primary and counter actors.

7.7 Summary

In this chapter we have introduced a *pattern template*. Each pattern consists of *Context*, *Problem*, *Solution*, *Example*, *Variations* and *Related Patterns* sections. In addition to textual descriptions, the context, problem and solution given by the patterns are represented using conceptual modelling techniques. We distinguish value and process viewpoints. We use e^3*value* to represent the value viewpoint and UML activity diagrams to represent the process viewpoint.

Furthermore, we have introduced a vocabulary, which describes the terminology used in the patterns. It includes two operating activities, *Primary* activity and *Counter* activity, and four control activities, *Witness*, *Verify*, *Promise*, and *Testify*. Every control activity considered produces a kind of *statement* as an output. This statement plays the role of an evidence document and can be used as an input to other control activities. In this way, activities can be *linked with each other* by matching inputs and outputs. For example, a witness activity produces a statement which can be used as a to-be-verified statement or a supporting statement of inputs for a *Verify* activity. In addition, the *Verify* and *Testify* activities have statements as a required input.

For each activity we define *control principles*, which describe permitted relationships between an activity and other activities, objects and actors, including relationships between a verify activity and an activity it verifies, a witness activity and an activity being witnessed, a promise activity and an activity it promises, and a testify activity and an activity it testifies. All the principles can be classified in one of the three groups: *Activities-Activities*, *Activities-Objects* and *Activities-Actors* principles. The principles are normative rules that are required to ensure that control weaknesses do not exist. Every pattern has to comply to these control principles.

Chapter 8

Procedural patterns

In this chapter we present a library of *procedural control patterns*. As has already been introduced in Chapter 6, we consider four procedural patterns: Partner Screening, Proper Contracting, Execution Monitoring, and Execution Confirmation.

We describe the patterns using the *patterns template*, as suggested in section 7.1. According to this template, each pattern consists of a *Context*, *Problem*, *Solution*, *Example*, *Variations* and *Related Patterns* section. In addition to textual descriptions, the patterns are represented using graphical conceptual modelling techniques. We distinguish between a value viewpoint, for which we use e^3*value* , and a process viewpoint, for which we use UML activity diagrams.

In a pattern, we consider two actors: a *primary* actor and a *counter* actor. In the problem of a pattern, we consider a counter actor who behaves *sub-ideally* and a primary actor who wishes to mitigate the loss caused by such behaviour. For the solution, we model control mechanisms introduced by the primary actor.

The value viewpoint of such a two-actor model does not change by introducing controls. The solution affects only the process viewpoint, and only impacts the value viewpoint if a primary or a counter actor delegates some activities to other actors. As was explained in Chapter 6 we consider delegation separately in the delegation patterns in Chapter 10 and here we consider only the two-actor exchange.

On the process level each solution should *comply with the control principles* described in the control principles vocabulary of section 7.6. The relevant control principles are described in short in the corresponding section within a pattern's solution. In addition, the principles are listed in detail in Appendix A.

8.1 Partner Screening

The **Partner Screening** pattern relates to verification of a counter actor's past performances before any commitment is made to him. So, a primary actor does not trust the counter actor to perform a future counter activity in an ideal way. There is a hidden characteristics problem: the counter actor may hide relevant information about his characteristics, such as skills, abilities, legitimacy (see section 2.1.2). One of the ways to obtain more certainty about a counter actor is to examine activities he performed in the *past* or to check if he has acceptable characteristics (e.g. size of the company, turnover, and reputation), see section 2.1.2. This pattern requires that a primary actor verifies the past activities or characteristics of a counter actor with respect to their compliance with stated requirements of ideality, which are assumed to be known by the primary actor. The graphical representation of the pattern is shown in Figure 8.1.

8.1.1 Context

A primary actor and a counter actor execute a *Primary* and a *Counter* activity, as a result of which they exchange value objects *PO* and *CO*. According to the Promising Principles in section 7.6, the primary actor confirms his commitment to execute the *Primary* activity before actually executing it. This is done by performing a *Promise* activity, which results in a *Confirmation Statement* being sent to the counter actor. The most common interpretation of the *Confirmation Statement* is a contract.

The *Promise* activity is added to the context, since the pattern is ex-ante. An ex-ante situation implies that a contract has not yet been signed by at least one of the parties. We, therefore, include the *Promise* activity, which models contracting process, into the context in order to show that the screening activities occur before the contract is signed.

8.1.2 Problem

A primary actor is not certain if a counter actor has the required abilities and the proper qualifications to execute the *Counter* activity. For example, a counter actor may not be able to deliver a product or a service of the required quality, may be dishonest, or have a reputation of not paying on time.

This sub-ideal situation is modelled with a sub-ideal value object *CO* in the value model and a liability token is assigned to the counter actor. This sub-ideal situation is expressed with the *Sub-ideal Counter* activity in the process model.

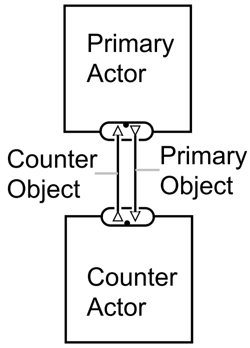
8.1.3 Solution

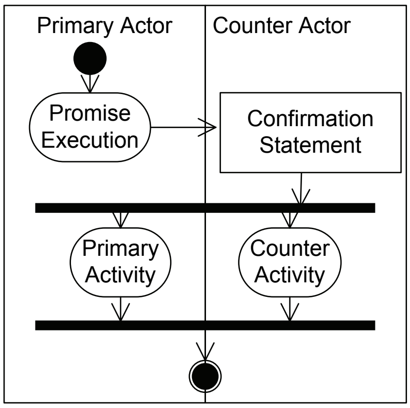
The primary actor must know what criteria the counter actor has to comply with. This can be legal criteria (e.g. a counter actor must have the right to import into a certain country) or commercial criteria (e.g. a counter actor should not have debts). Then the primary actor must verify if the past activities of the counter actor demonstrate compliance with these criteria. The compliance of past activities with the criteria will create reasonable certainty about the performance of the future counter activity.

In the solution of the pattern in Figure 8.1 this mechanism is modelled in the following way. Before making any commitments to the counter actor by executing a *Promise* activity, the primary actor must first verify known past activities of the counter actor or their results (see *Past Counter Activity*). The *Verify* activity checks whether the past execution or results of the counter

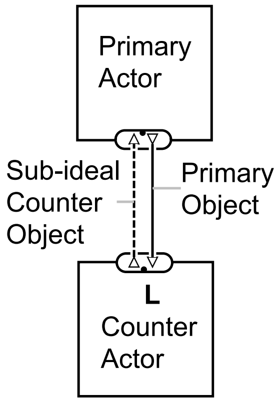
Name: Partner Screening

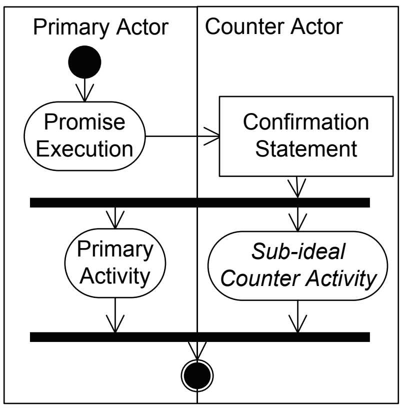
Context



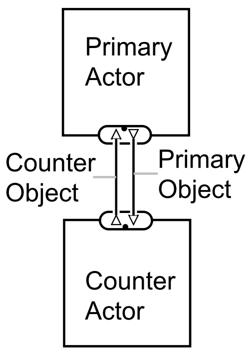


Problem





Solution



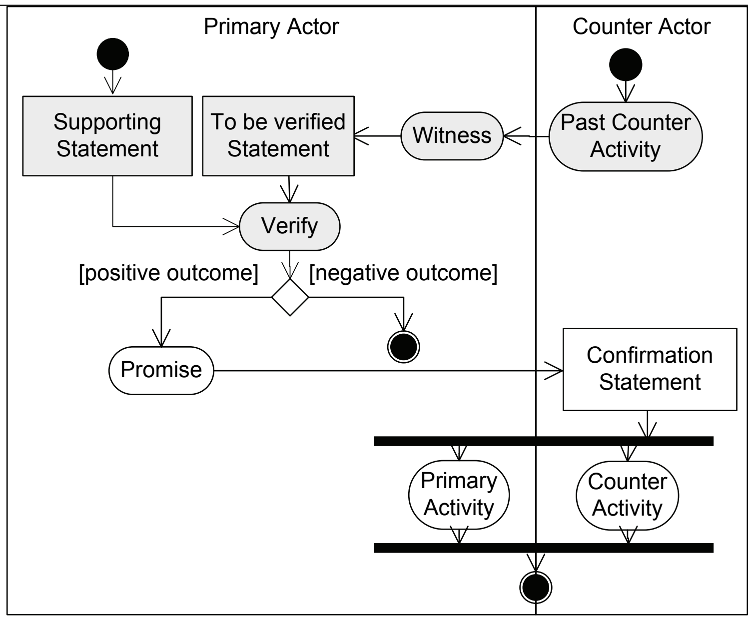


Figure 8.1: Pattern ‘Partner Screening’

activities comply with the primary actor's criteria, represented in the *Supporting Statement*. Furthermore, the primary actor has to witness the past activities or their results before executing the verification activity. The outcome of the *Witness* activity is the *To-be-verified Statement*.

The contracting activity *Promise* and the further execution of the *Primary* and *Counter* activities only take place if verification shows that the *Past Counter Activity* satisfies the criteria in the *Supporting Statement*.

If verification is negative, the *Primary* and *Counter* activities are not executed. Such an outcome corresponds to a value transfer with two empty value objects and is normally not modelled.

Control Principles

The control principles, which are relevant to the pattern Partner Screening, are Verification, Witnessing, Promising, and Pre-execution (see section 7.6). Among other things, they require witnessing to be performed after the past counter activity and before verification. Further they require the direct transfer of the supporting document. Finally, they require verification and generation of supporting statements to be carried out by an actor who is independent and socially detached from the counter actor. The segregation of activities is modelled by means of swim lanes, as was explained in Chapter 7.

The Pre-execution principles of the pattern Partner Screening requires the *Verify* activity to be performed before the *Promise* activity. This is because it makes more sense to screen a counter actor before a contract is signed so that to prevent engaging in a relationship with a dubious party. This requirement can be relaxed if the primary actor trusts the counter actor. For a more precise description of the principles see the Appendix A.

8.1.4 Example

We apply the pattern Partner Screening to an exchange between a buyer and a seller, shown in Figure 8.2. As considered here, the buyer's control problem is that the seller may not deliver goods on time. Similar problems are an unreliable seller, a seller who sells products or services of bad quality, or even a seller who is not eligible to deliver in a particular region or who is not eligible to sell particular services. For the particular problem of untrustworthy delivery times the solution requires the buyer to check if previous deliveries made by the seller were on time.

The instantiation of the context is represented by an ideal process model in Figure 8.2. Here, the buyer is a primary actor, the seller is a counter actor, *Pay Goods* is a primary activity, *Deliver Goods* is a counter activity, *Send Order* is a promising activity, and *Purchase Order* is the confirmation statement. The control problem is represented by a sub-ideal process model in Figure 8.2, where the *Deliver Late* activity is the sub-ideal counter activity.

In the solution process model in Figure 8.2 the buyer should witness previous deliveries made by the seller and compare information about the seller's previous delivery times in *Receiving Report*, with agreements about the delivery time in *Delivery Terms*. If such a comparison reveals that most of the deliveries made by the seller were on time (see *[positive outcome]*), the buyer

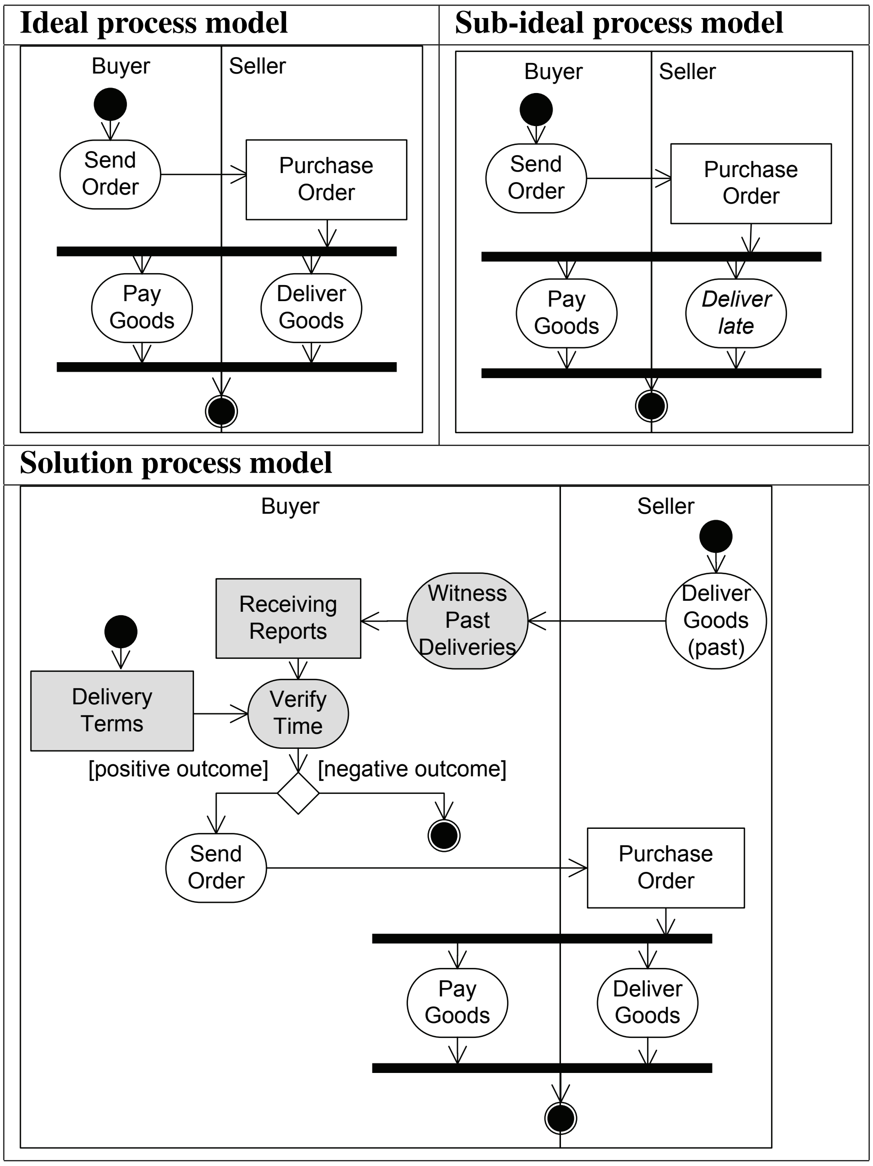


Figure 8.2: Example of Partner Screening pattern

should proceed with sending the *Purchase Order*. Otherwise, the buyer is not recommended to order from this seller.

If the buyer has not collected information on past deliveries made by the seller (e.g. because the buyer had never traded with the seller), then he can obtain such information from a trusted party. This can be modelled using the delegation patterns from Chapter 10.

8.1.5 Variations

No Witness activity. A common practice when selecting a counter party is not to judge his past performance, but to assess his characteristics, such as size of the company, popularity of brand, reputation, and turnover. For example, a primary actor may believe that large organizations deliver better service, in which case he may select the counter actor based on the counter actor's profits, number of employees, and other similar criteria. Such a control is known as signalling (see section 2.1.2), which can be modelled as a variation of the solution of the pattern Partner Screening, see Figure 8.1. In such a variation, the *To-be-verified Statement*, or selection criteria, is no longer an outcome of the *Past Counter* activity, so this activity disappears. The *Witness* activity then refers to the witnessing of information about these criteria. An obvious additional requirement is that the *To-be-verified Statement* is generated by a trustworthy source.

Witnessing of behaviour. In the pattern in Figure 8.1 we consider *witnessing of output* of the *Past Counter* activity. As stated in the Witnessing principle in section 7.6, the witnessing of output requires the direct transfer of the output of the witnessed activity to the witness activity. This is to say, the output of the activity must unambiguously determine the *Past Counter* activity. If this is not possible, the witnessing activity must occur *simultaneously* with the *Past Counter* activity. This should be modelled according to the *Witnessing of Behaviour* principle described in section 7.6.

No Promise activity. In some cases, the *Promise* activity is absent, for example, it is not performed at all or it is just not included into the model because it is not relevant to the study. In such cases, it should be ensured that verification precedes the *Primary* activity.

8.1.6 Related patterns

The problem described in the pattern Partner Screening can also be treated by the patterns Penalty, Incentive, and Execution Monitoring. The difference in Partner Screening is that it is an ex-ante control which is performed before a contract is signed.

8.2 Proper Contracting pattern

The **Proper Contracting** pattern describes how the process of contracting must be performed to avoid misunderstandings in the future. The associated control problem is that a counter actor refuses to recognize that he made a commitment to a primary actor and, as a result, refuses to execute the counter activity. The control mechanism requires the counter actor to provide documentary evidence of his commitment to the primary actor. This is normally accomplished by getting the counter party to sign a contract.

8.2.1 Context

A primary actor and a counter actor execute a *Primary* and a *Counter* activity, as a result of which they exchange value objects *PO* and *CO*. The primary actor confirms his obligation to execute a *Primary* activity before actually executing it. He confirms his commitment by performing a *Promise* activity, which results in a *Confirmation Statement* being sent to the counter actor.

The *Promise* activity is added to the context, since the pattern is ex-ante. An ex-ante situation implies that a contract has not yet been signed by at least one actor. We, therefore, include the *Promise* activity, or contracting process, into the context in order to model that the screening activities occur before the contract is signed.

8.2.2 Problem

A counter actor does not confirm his commitments to a primary actor, as a result of which, the counter actor may deny any commitments made to the primary actor and refuse to execute the counter activity in return for the primary activity or not execute the counter activity as agreed (i.e. sub-ideally).

On the value level, this situation is modelled with the sub-ideal value transfer of *CO*. On the process level, it is modelled with the sub-ideal activity *Sub-ideal Counter* activity. In addition, the counter actor executes the *Deny Commitment* activity, which denotes a denial action.

8.2.3 Solution

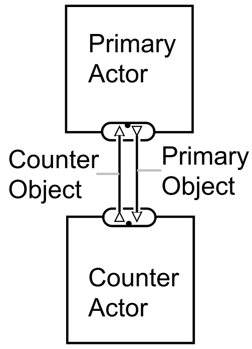
The problem from the primary actor's perspective is that he cannot prove that the counter actor made the commitment to execute the counter activity. The solution is to create such a proof by introducing an evidence document.

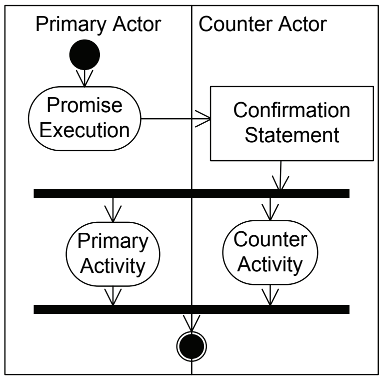
The solution in Figure 8.3 suggests that, before making any commitments, the primary actor must obtain a conformation statement from counter actor, in which a promise to execute the counter activity is made. Thus, the counter actor must execute an activity *Promise (CA)* and send a *Confirmation Statement (CA)* to the primary actor [1]. The *Confirmation Statement* is evidence of the counter actor's commitment to execute the *Counter* activity in return for the *Primary* activity. If the counter actor later denies his commitment, this *Confirmation Statement* can be used to disprove it.

[1] CA indicates that the activity is executed by the *C*ounter *A*ctor

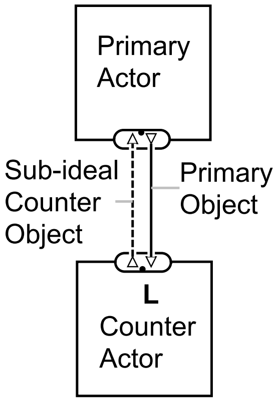
Name: Proper Contracting

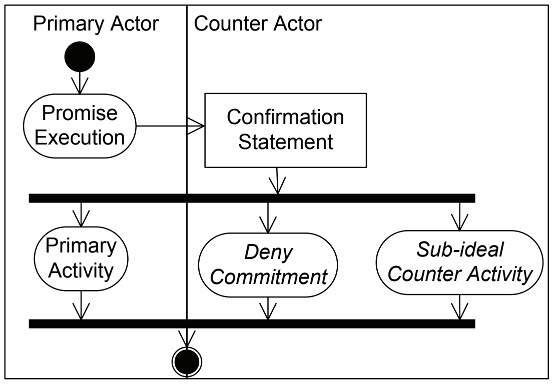
Context



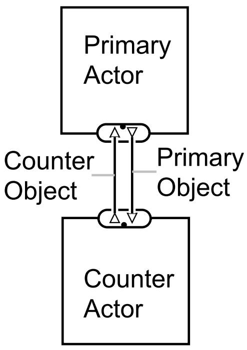


Problem





Solution



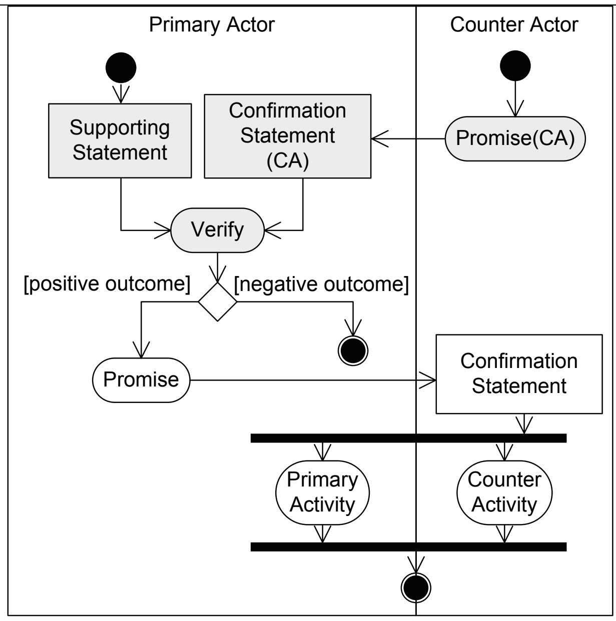


Figure 8.3: Pattern ‘Proper Contracting’

Furthermore, the primary actor uses a *Supporting Statement* to verify the confirmation statement with respect to its completeness, correctness and legitimacy. Here, the *Supporting Statement* represents the criteria laid down the primary actor with respect to what should be in the *Confirmation Statement*. The *Confirmation Statement* is a contract between counter and primary actors, and the criteria to which it must comply are to be found in the *Supporting Statement* which may represent regulations that a primary actor is bound to, e.g. a requirement of a contract to be signed by the chairman of the counter actor's company, and not just by a middle-level manager.

If the outcome of the verification is positive, the primary actor may continue with making his promises to the counter actor by sending the *Confirmation Statement*. Otherwise, the process stops and the *Promise*, *Primary* and *Counter* activities are not executed. Such an outcome, omitted in the value model, corresponds to a value transfer of two empty value objects.

Control Principles

The control principles relevant to this pattern are Verification, Promising, and Pre-execution. They require the primary actor to execute the primary activity only after a promise to execute it is made. They also require the primary actor to execute a *Promise* activity only after verification of promises made by a counter actor. The segregation of activities is modelled by means of swim lanes, as was explained in Chapter 7. In addition, the direct transfer of the supporting statement is required.

The Pre-execution principles of the pattern Proper Contracting require the counter actor to make the promise first. This requirement can be relaxed depending on the level of trust between the two actors. Normally, the contracts are signed simultaneously. For a more precise description of the principles see the Appendix A.

8.2.4 Example

We apply the pattern to an exchange between a buyer and a seller, shown in the ideal process model in Figure 8.4. In this model, a seller sends a buyer a *Quote* of prices for certain goods. In the ideal and sub-ideal models we assume that if the buyer does not respond to the quote, he agrees to buy the goods. In fact, by sending the quote the seller makes a commitment to deliver the goods. The control problem in such a case is that when the seller delivers the goods, the buyer may refuse to pay the price quoted.

The solution suggests that the seller should not make the commitment to deliver the goods for a certain price, unless the buyer confirms his agreement to pay it. In accordance with the pattern Proper Contracting, we add an activity *Order* to the buyer which is an instance of the pattern's activity *Promise (CA)*. By performing the activity *Order*, the buyer sends a confirmation statement *Purchase Order*, in which he states the price he is ready to pay for the goods. The seller then verifies if the price stated in the purchase order is the same as in the quote, and, if so, sends an order acknowledgement and delivers the goods. Otherwise, the goods are not delivered.

Note that in the solution model the primary actor makes a commitment to deliver the goods only by sending the *Order Acknowledgement*, which is different from the sub-ideal model, in which a commitment is already made by sending the *Quote*.

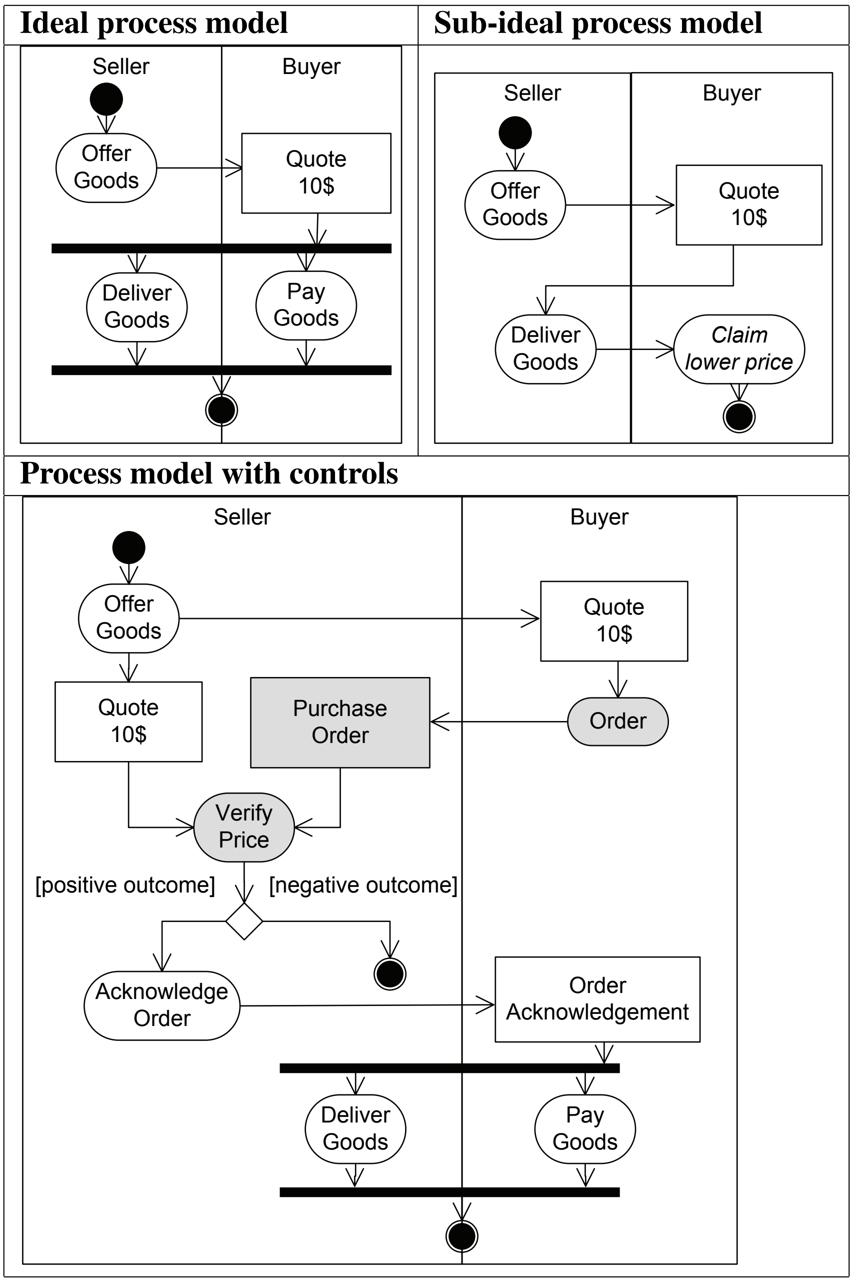


Figure 8.4: Example of Proper Contracting pattern

8.2.5 Variations

Self-confirmation. A usual variation of this pattern is when a counter actor's obligation is created as a result of his non-response to the *Confirmation Statement*. The primary actor sends a

Confirmation Statement (e.g. contract) to the counter actor and gives him some time to respond. By responding, the counter actor can *cancel* the contract and thus prevent the binding. However, if the counter actor does not respond, he finds himself bonded by the contract. So, the counter actor's non-responsiveness during the allotted period of time is equivalent to his execution of the *Promise* activity.

8.3 Execution Monitoring pattern

The **Execution Monitoring** pattern describes the control problem in which a counter actor does not execute his commitments or executes them in a sub-ideal way (e.g. not as agreed in the contract). The control mechanism requires the primary actor to monitor counter activity. This solution is very similar to the Partner Screening pattern, the only difference being that verification concerns the counter activities under the contract and not an actor's past counter activities.

8.3.1 Context

A primary actor and a counter actor execute a *Primary* and a *Counter* activity, as a result of which they exchange the value objects *PO* and *CO*. This pattern is ex-post, which means that the commitments between the counter and the primary actor have already been established.

8.3.2 Problem

The primary actor is not certain if the counter activity complies with certain requirements, e.g. the agreements made in the contract, some legislation or some generally-accepted norms. This may lead to doubts both with regards to the proper *execution* of the activity or with regards to its *outcome*.

This sub-ideal situation is modelled with a sub-ideal value object *CO* in the value model and is expressed with the *Sub-ideal Counter* activity in the process model. The liability token is assigned to the counter actor in the value model.

8.3.3 Solution

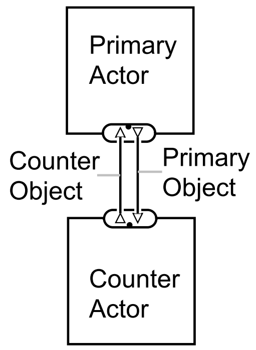
The primary actor must verify if the counter activity or its outcome comply with the requirements, as stated in a *Supporting Statement*. Only if compliance is established should the primary actor execute the *Primary* activity.

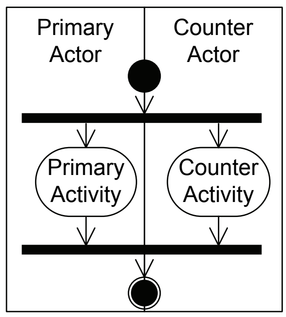
This is modelled as follows in Figure 8.5. Before executing the *Primary* activity, the primary actor performs verification of the outcome of the *Witness* activity, which observes the counter activity.

The outcome of the *Witness* activity is a *To-be-verified Statement*, which contains information about the counter activity. The *Verify* activity compares the information in the *To-be-verified*

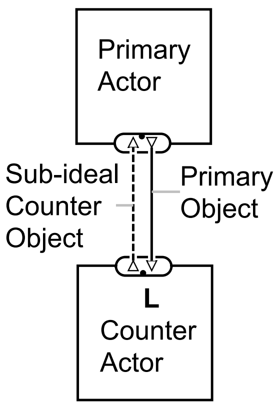
Name: Execution Monitoring

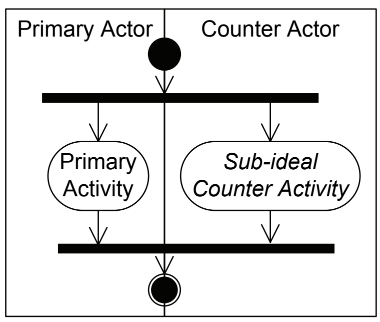
Context



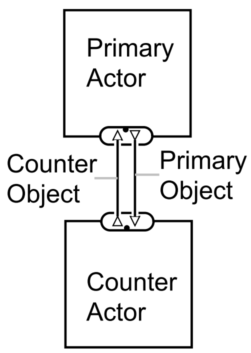


Problem





Solution



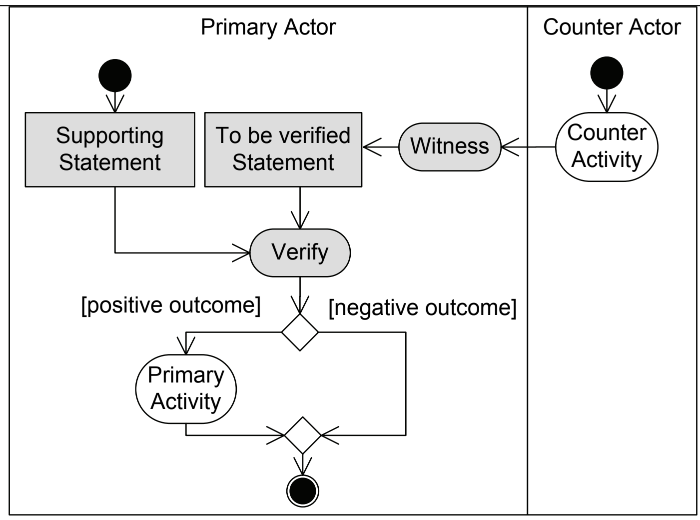


Figure 8.5: Pattern ‘Execution Monitoring’

Statement and the *Supporting Statement*. The primary actor should only execute the *Primary* activity if the outcome of verification shows that the *Counter* activity complies with the previous agreements stated in the *Supporting Statement*.

Control Principles

This control involves the Verification, Witnessing and Pre-execution principles (see section 7.6). Among other things, they require the *Counter*, *Witness*, and *Verify* activities to be executed one after another. Furthermore, they require the *Supporting Statement* to be transferred directly to the *Verify* activity from a trustworthy source. Finally, they require a proper segregation of duties, e.g. execution of verification and witnessing by a party who is not acting in the interests of the counter actor. The segregation of activities is modelled by means of swim lanes, as was explained in Chapter 7.

The Pre-execution principles for the pattern Execution Monitoring require verification to be performed before the primary activity. For example, a buyer should not to pay before he checks the delivered goods. This requirement, however, can be relaxed if the primary actor trusts the counter actor, e.g. if the buyer knows that the seller always delivers the goods (e.g. from previous experience). For a more precise description of the principles see the Appendix A.

8.3.4 Example

We apply the pattern to an exchange between a buyer and a seller, shown in Figure 8.6. The control problem is that the seller may deliver goods which were not ordered by the buyer. A similar problem is that the seller does not deliver goods at all, delivers them too late, delivers damaged goods, etc. The control mechanism, according to the pattern, requires the buyer to pay only if reconciliation with the purchase orders proves that the delivered goods have been ordered.

The instantiation of the context is represented by an ideal process model: *Pay Goods* is a primary activity, and *Deliver Goods* is a counter activity. The control problem is represented by a sub-ideal process model, where *Deliver Wrong Goods* is the sub-ideal counter activity.

The process model of the solution shows that the buyer should pay for the goods only after delivery is reconciled with the corresponding purchase order. So, the buyer executes *Pay Goods* after the activity *Verify Delivery*. This verification means checking the correctness of quantity, quality and legitimacy of the delivery using a supporting statement, which is a *Purchase Order* in this case. Verification also requires witnessing the delivery. The outcome of the *Witness Delivery* activity is a *Receiving Report*, which is used as the to-be-verified statement.

8.3.5 Variations

No Witness activity. The outcome of the counter activity can be a statement, e.g. when the counter activity is a reporting activity. Then, witnessing refers to witnessing of this statement. In such cases, the *Witness* activity can be omitted, since the activity diagrams assume that the receiving party possesses the document thereby witnessing it.

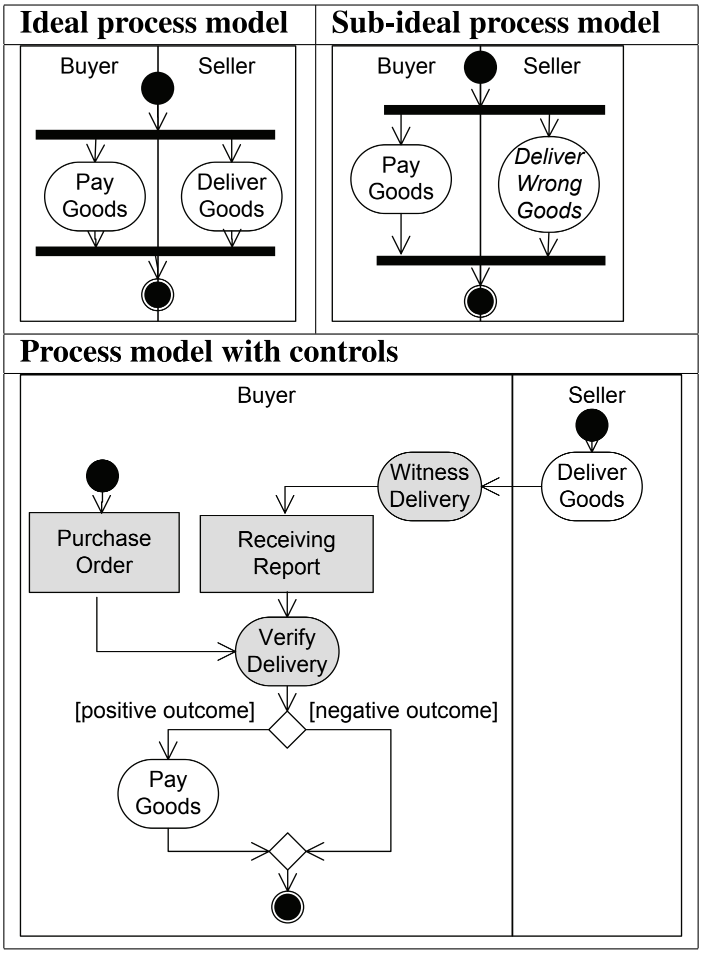


Figure 8.6: Example of Execution Monitoring pattern

Witnessing of behaviour. Here we consider *witnessing of output* of the *Counter* activity. As stated in the Witnessing principle in section 7.6 output should unambiguously determine the *Counter* activity, e.g. it should contain correct information which represents the reality. If output has no such characteristics, the witnessing activity must occur *simultaneously* with the *Counter* activity. This should be modelled according to the *Witnessing of Behaviour* principle described in section 2.1.1.

Self-reporting statement. Another frequent situation in practice is when a counter actor reports a counter activity himself. We call this **self-reporting**. For example, the counter actor is a beer producer who sends an invoice stating how many bottles of beer he has delivered. This self-reporting statement – the invoice – should not be used as the *To-be-verified Statement*, because it is not generated by the primary actor or his trusted party and, thus, may contain incorrect information, e.g. an overstated or understated number of bottles. From the perspective of the Execution Monitoring pattern, this self-reporting statement can be eliminated, since it is not needed for the controls described here. However, a self-reporting statement is needed for the controls described in the Execution Confirmation pattern, as shown in the 'Variation' section of the Execution Confirmation pattern in section 8.4.

8.3.6 Related patterns

The problem described in the pattern Execution Monitoring can also be treated by the patterns Penalty, Incentive, and Partner Screening. The specific characteristic of Execution Monitoring is that it considers ex-post procedural control.

8.4 Execution Confirmation pattern

The **Execution Confirmation** pattern describes a control problem in which a counter actor incorrectly claims that the primary actor executed a primary activity sub-ideally. The control mechanism suggests introducing an evidence document, in which the counter actor testifies about the execution of the primary activity. The primary actor can use such a testifying document later in (legal) disputes as proof that the counter actor agreed to the ideality of the primary activity.

8.4.1 Context

A primary actor and a counter actor execute a *Primary* and a *Counter* activity, as a result of which they exchange the value objects *PO* and *CO*. This pattern is ex-post, which means that the commitments between the counter and the primary actor have already been established.

8.4.2 Problem

A primary activity is executed ideally. However, a counter actor claims that that is not the case and either refuses to execute the counter activity or, if the counter activity has already been executed, requires compensation. As a result, the primary actor does not get the counter value object *CO*. The value transfer of *CO* is sub-ideal.

On the process level, the counter actor executes the *Counter* activity sub-ideally. In addition, he executes an activity *Deny Execution*, which models his claim about the sub-ideality of the *Primary* activity.

8.4.3 Solution

The primary actor must ensure that the counter actor testifies about the primary activity in a document, which is unconditional evidence of the ideal execution of the *Primary* activity and can be used later in a legal dispute, should the counter actor deny this.

The counter actor will always check if the *Primary* activity is ideal before testifying to it. We consider that the counter actor witnesses the *Primary* activity and verifies its ideality, even though such verification is not required by the primary actor. The witnessing is done at the same time as the primary activity. Here we consider witnessing the execution of the primary activity. The witnessing of its outcome is also possible and should be modelled according to the Witnessing principles described in section 2.1.1.

The *Supporting Statement*, used in verification, represents previous agreements on the ideality of the *Primary* activity. The *To-be-verified Statement* states the results of the counter actor's witnessing of the *Primary* activity. After the verification, the counter actor can do two things. If verification reveals that the *Primary* activity is ideal, then the counter actor testifies about it in a *Testifying Statement*. Otherwise in case of *[negative outcome]*, the counter actor denies the ideality of the primary activity by executing a *Deny Execution* activity.

Note that this pattern contains the elements of the pattern Execution Monitoring, applied from the counter actor's perspective. Unlike in other patterns, here we also take the counter actor's viewpoint into account by assuming that he does not trust the primary actor and will not testify about the primary activity before verifying it. This verification is not needed from the primary actor's perspective. However, it is not realistic to assume that the counter actor will testify about the primary activity without checking it. Therefore, the counter actor's perspective is also taken into account.

Control Principles

The solution should observe the Verification, Witnessing, Testification, and Pre-execution control principles (see section 7.6). Among other things, they require the *Witness and Verify* activities to be executed one after another. The *Witness* and *Testify* activities should be executed at the same time as the *Primary* activity. Here, the Witnessing of Behaviour principles are applied (see section 7.6). In fact, a primary actor would like to receive the testification statement before a primary activity is executed. However, this will always conflict with the counter actor's

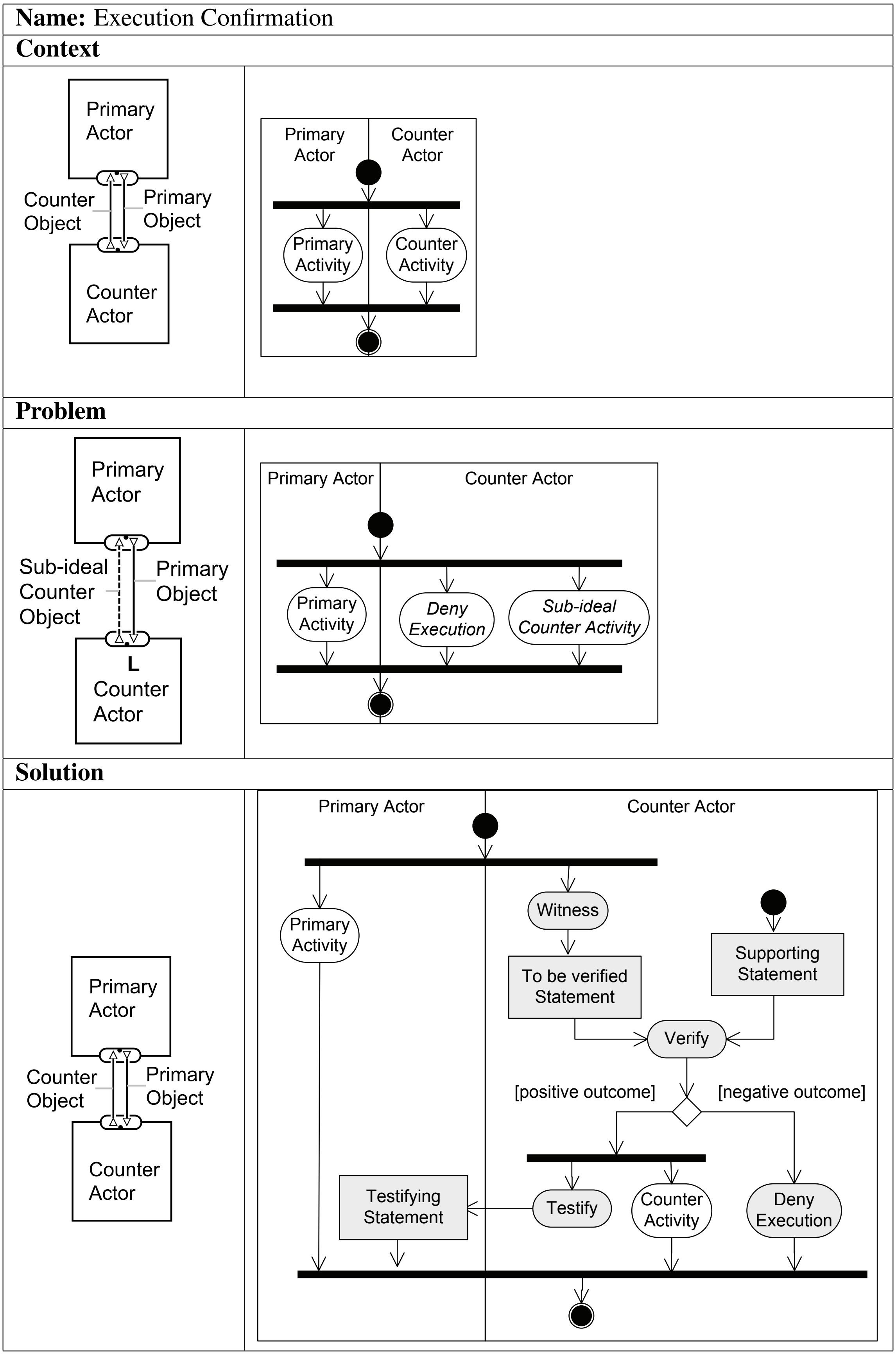


Figure 8.7: Pattern 'Execution Confirmation'

perspective: according to the Testifying principles, the counter actor must not testify about the primary activity before it is executed. An example of simultaneous execution is a carrier who delivers goods to their destination, but does not trust the recipient. The receipt issued as evidence of delivery is handed over in return for goods, which requires that both parties are present during the exchange of goods and the receipt of them.

Furthermore, the principles require the *Supporting Statement* to be transferred directly to the *Verify* activity from a source that is independent of the primary actor. A proper segregation of duties is required, e.g. execution of verification and witnessing by a party not acting in the interests of the primary actor. The segregation of activities is modelled by means of swim lanes, as was explained in Chapter 7. Finally, the *Testifying Statement* also has to be transferred directly to the primary actor.

The Pre-execution principles of the pattern Execution Confirmation require the execution of the *Verify* activity before the *Testify* activity. The requirement, however, can be relaxed, if the counter actor trusts the primary actor. For a more precise description of the principles see the Appendix A.

8.4.4 Example

As an example, we apply the solution of the pattern to an exchange between a buyer and a seller as shown in Figure 8.8. The control problem involved is the seller's claims that the buyer has not paid for goods. To solve the problem, the control mechanism requires the buyer to receive a receipt from the seller. The receipt can always provide evidence of payment, if later on the seller claims that the buyer has not paid.

In the ideal process model, the activity *Pay Goods* is a primary activity, and the activity *Deliver Goods* is a counter activity. The control problem is represented by a sub-ideal process model. *Claim Less Payment* is an instance of the *Deny Execution* activity.

In the solution process model, the seller testifies the buyer's payment by issuing a receipt after witnessing and verifying payment with actual delivery. Here the *Delivery Report*, used as the supporting statement, states what the payment *should be*. The *Payment Record* plays the role of the *To-be-verified Statement* and states what the payment actually *is*.

8.4.5 Variations

Self-confirmation. A frequent variation of this pattern is when the primary actor is empowered to use his own invoice or a similar document as evidence of execution of the *Primary* activity. This is different from requiring the counter actor to testify, as described in the pattern in Figure 8.7. In this variation, the primary actor sends a statement to the counter actor, by which he makes a claim about the execution of the primary activity. The counter actor then has some time to react. If he does not deny what is claimed in the statement, it will have the same power as the *Testifying Statement* otherwise issued by the counter actor.

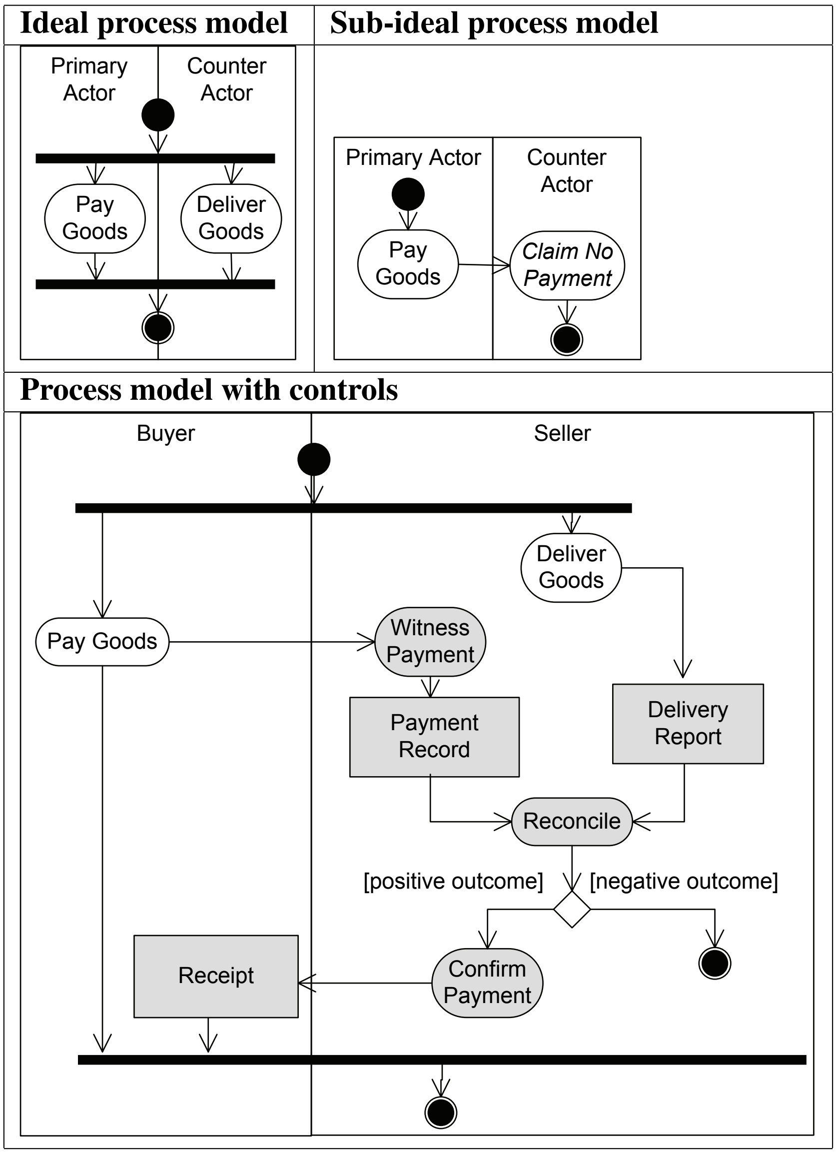


Figure 8.8: Example of Execution Confirmation pattern

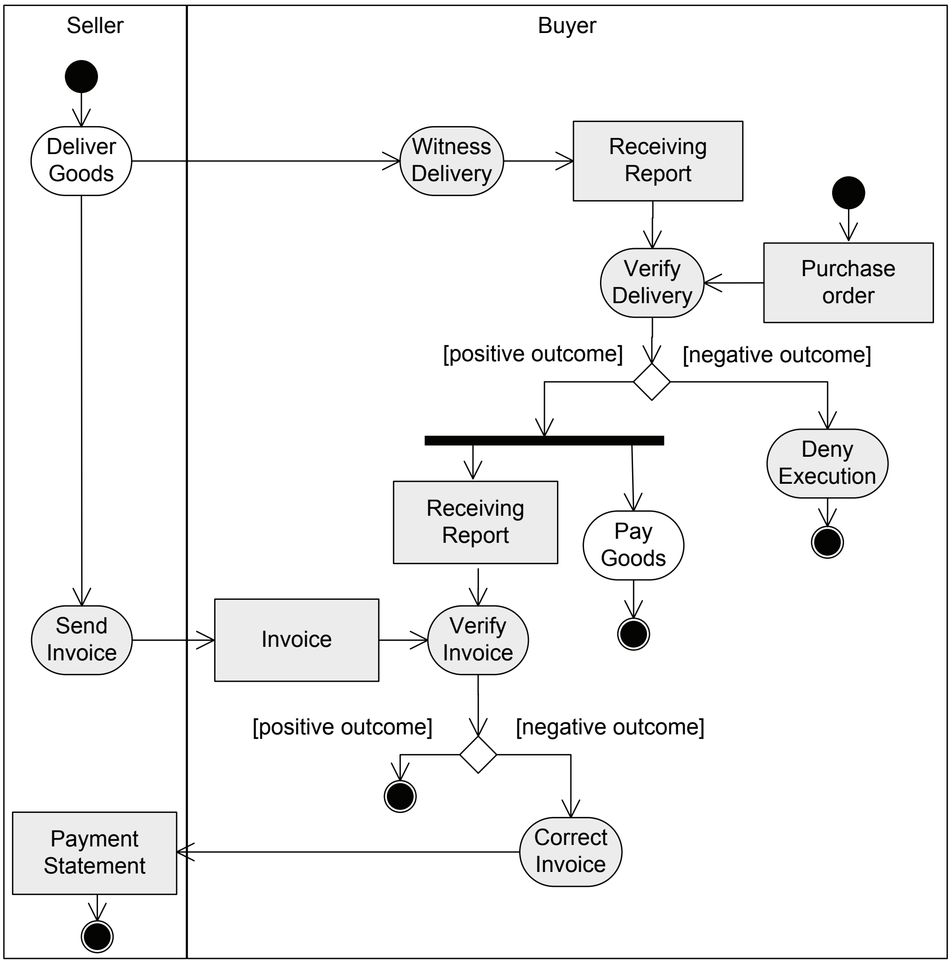


Figure 8.9: Example of the variation of the Execution Confirmation pattern

For example, in Figure 8.9 we have a buyer as a counter actor and a seller as a primary actor. The seller sends the buyer an invoice on which he states that ten containers of beer have been delivered and the buyer has a term of three months to claim the opposite. According to the pattern in Figure 8.7, before paying the invoice, the buyer first verifies that a delivery, which corresponds to the purchase order, has been made and, second, he verifies if the invoice corresponds to the delivery. In the example, the buyer pays if ten containers of beer have been delivered as stated in invoice. If only eight containers of beer have been delivered, the buyer pays for them, and sends a correction note *Payment Statement* to the seller which states that only eight containers were delivered. However, if the buyer does not send the *Payment Statement* within the three month term, the seller can claim that the buyer testified the delivery of the ten containers and is obliged to pay for all ten of them.

So, if the invoice is correct and the buyer does not send the seller any corrections, the copy of the invoice can be used by the seller to prove that the buyer is satisfied with the delivery. If the invoice is incorrect and the buyer sends a correction note, the note then plays the role of the testifying statement.

8.5 Summary

In this section we have introduced four procedural patterns: Partner Screening, Proper Contracting, Execution Monitoring and Execution Confirmation. The Partner Screening and Execution Monitoring patterns describe the monitoring of a counter actor's activities. The difference between these two patterns is that Partner Screening suggests an ex-ante control, and Execution Monitoring suggests ex-post control. The pattern Proper Contracting describes how the contracting process should be arranged to avoid misunderstandings about the commitments the actors make to each other. Finally, the pattern Execution Confirmation suggests how the final process of the transaction should be arranged to avoid misunderstandings about the activities performed.

The control mechanisms suggested in the procedural patterns affect only the process viewpoint. These controls are implemented by adding new operational control activities and objects, such as evidence documents which are not necessarily of value. Therefore, the value perspective does not change. A change in the value perspective can occur when some of the added control activities are delegated to a third party as will be described in Chapter 10.

Chapter 9

Contractual patterns

In this chapter we describe the Incentive and Penalty patterns of contractual controls. In an Incentive pattern, a primary actor gives a counter actor a reward in order to refrain him from potential sub-ideal behaviour. In the Penalty pattern, a primary actor introduces a penalty to punish a counter actor for behaving sub-ideally.

Each pattern consists of a *Context*, *Problem*, *Solution*, *Example*, *Variations* and *Related Patterns* sections which are similar to the procedural patterns described in Chapter 8. As in procedural patterns, we distinguish between a value viewpoint, for which we use e^3*value* , and a process viewpoint, for which we use UML activity diagrams. Furthermore, in Chapter 7 we introduced a *vocabulary*, which describes the terminology used in the patterns. Each pattern contains elements described in the vocabulary.

The process models of the Incentive and Penalty patterns are variations of the Execution Monitoring pattern. The difference is that these patterns incorporate value-based control instruments – a penalty and an incentive, which stimulate the counter actor's ideal behaviour. Compared to process models, value models provide us with a better model for the analysis of penalties and incentives, because they allow us to reason about the size of the monetary incentive using the e^3*value* concepts of objective and subjective value.

9.1 Penalty pattern

In Figure 9.1 illustrates a Penalty pattern. Below we discuss its context, problem and solution.

9.1.1 Context

Value view. A primary actor and a counter actor execute a Primary and a Counter activity, as a result of which they exchange the value objects *PO* and *CO*.

Process view. At the process level, the value transfers *PO* and *CO* correspond to the execution of a primary activity and a counter activity correspondingly.

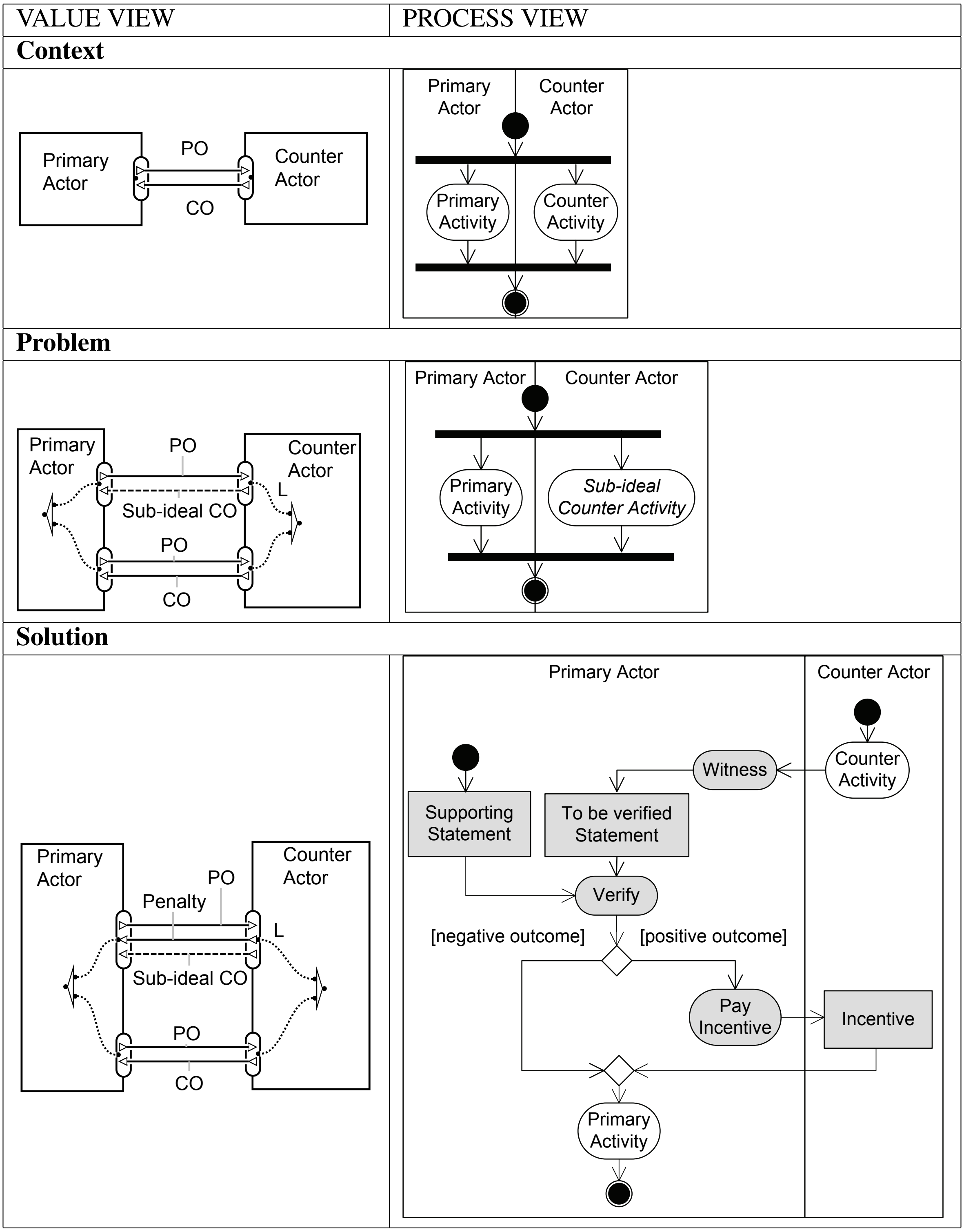


Figure 9.1: Penalty pattern

9.1.2 Problem

Value view. The sub-ideal value model shows that a counter actor has a choice behaving ideally or sub-ideally, which is modelled with two dependency paths: the ideal path and sub-ideal path. The value transfers of the *ideal path* are the same as in the context model: *PO* and *CO*. In the *sub-ideal path*, the counter actor transfers a sub-ideal counter value object *Sub-ideal CO*, while the primary actor transfers the ideal value object *PO*. In addition, a liability token **L** has been assigned to the counter actor to indicate that he is responsible for the sub-ideality.

Although we depict the *ideal* value transfer *PO* in the models, this pattern also covers a situation in which *PO* is sub-ideal. For example, if a seller delivers damaged goods, a buyer can (1) pay for them, which corresponds to the transfer of the ideal value object *PO*, or (2) refuse to pay for them, which corresponds to the transfer of the sub-ideal value object *Sub-Ideal PO*. The solution given by the pattern is applicable to both situations. Further we consider only the situation when *PO* is ideal.

Process view. In the process view, the transfer of the *Sub-ideal CO* corresponds to a *Sub-ideal Counter* activity. For brevity of representation, in the process model in Figure 9.1 we show a process that only corresponds to the sub-ideal path.

9.1.3 Solution

Value view. The solution suggests that to prevent a counter actor from taking the sub-ideal path, the value he accumulates in the sub-ideal path should be *lower* than the value he accumulates in the ideal path. This is achieved by requiring the counter actor to transfer an outgoing value object *Penalty* in the case of sub-ideal behaviour, as shown in the solution part of Figure 9.1.

The value object *Penalty* is included into the same value interface as the value object *Sub-ideal CO* and is transferred from the counter actor to the primary actor. This models that as soon as the counter actor transfers *Sub-ideal CO*, he has to transfer the value object *Penalty*.

There are also other ways of modelling a penalty in e^3*value*, for example, as a value object in another value interface of the sub-ideal path. The main requirements for the penalty value object is that (1) it is added to the path that corresponds to the sub-ideal behaviour of the counter actor, and (2) it reduces the *value* the counter actor accumulates in the sub-ideal path. This last point will now be further discussed in more detail.

Penalty size. Actors accumulate value by exchanging value objects. A counter actor chooses the sub-ideal path if the value he accumulates in this path is higher than the value he accumulates in the ideal path. The goal of the penalty in Figure 9.1 is to make the value a counter actor accumulates in the sub-ideal path *lower* than the value he accumulates in the ideal path. If we assume that all the value objects in Figure 9.1 are monetary, the following condition should hold:

$$PO - CO > PO - SubIdeal\ CO - Penalty. \quad (9.1)$$

This states that the value accumulated by a counter actor in the ideal path is higher than the value he accumulates in the sub-ideal path.

However, the objects *PO*, *CO* and *Sub-ideal CO* are not always monetary. One of the two value objects exchanged in a value transaction usually represents goods and services and the other one represents money paid in return. According to e^3*value* in Chapter 3, an important point that has to be taken into account is that actors value non-monetary objects subjectively, e.g. depending on their wealth, surroundings and needs. In economics this incorporated into concepts of *utility* and *budget constraints* [Varian, 2006].

Such a subjective valuation influences the decision about the size of a penalty in the condition 9.1. A penalty of a fixed amount will be effective for some actors, but not for others. For example, a person in a hurry will not consider a penalty of 100 Euro for parking illegally in the centre of Amsterdam so high as compared to the time he spends for looking for a legal parking space. A less busy person will consider this penalty high and will spend time looking for a parking space to avoid the penalty.

The subjective values of the non-monetary value objects can also be expressed in monetary units. For example, we can say that a busy person values the time spent on looking for a parking space as a loss of 1000 Euro. The penalty of 100 Euro will not compensate the loss of 1000 Euro, so this person will choose to pay the penalty and park illegally, thereby saving time. The less busy person, who values the time at 10 Euro does not wish to pay the penalty, because he will then make a loss of 90 euro. To sum up, the penalty is not effective for a busy person, but it does have an effect on a less busy person.

As a result, we rewrite the criterion for the effective penalty size as follows:

$$PO_{CA} - CO_{CA} > PO_{CA} - Sub - ideal\ CO_{CA} - Penalty. \quad (9.2)$$

By using the lower index *CA* we indicate here that the objects *PO*, *CO* and *Sub-ideal CO* are valued from the subjective perspective of the *C*ounter *A*ctor (*CA*). Note that we assume that the value objects *PO*, *CO* and *Sub-ideal CO* are non-monetary and the *Penalty* is monetary. Thus, the penalty amount depends on how the counter actor values the objects *PO* and *CO* and *Sub-ideal CO*.

Besides the counter actor's perspective, we should also take that of the primary actor into account. A primary actor wishes to get a refund or even extra compensation for a loss caused by a counter actor. So, the primary actor wants the accumulated value in the sub-ideal path to be at least the same as the accumulated value in the ideal path. This can be formulated as follows:

$$Sub - ideal\ CO_{PA} + Penalty - PO_{PA} \geq CO_{PA} - PO_{PA}, \quad (9.3)$$

in which the left-hand side of the inequality represents the value accumulated by the primary actor in the sub-ideal path, and the right-hand side represents the value accumulated by the primary actor in the ideal path. By using the lower index PA we indicate that the objects *PO*, *CO* and *Sub-ideal CO* are valued from the subjective perspective of the *P*rimary *A*ctor (*PA*). The penalty is assumed to be monetary, while other *PO*, *CO* and *Sub-ideal CO* are non-monetary.

Penalty size in complex cases. The models can be more complicated in real-life cases. An ideal and/or sub-ideal path could contain more than one value transfer providing there are AND- or OR-forks, or a value transfer could contain more than one object *PO*, *CO* or *Penalty*. In such cases, the calculation of the accumulated value should take into account these additional value objects. A general criterion for an effective penalty is to make the subjective value, which the primary (counter) actor accumulates in the ideal path, at least the same as (or higher than) the subjective value he accumulates in the sub-ideal path.

Process view. At the process level, the solution requires verification of the ideality of a counter activity before penalizing a counter actor, see Figure 9.1. This solution is a variation of the solution of the Execution Monitoring pattern, which was discussed in detail in section 8.3. In short, a primary actor has to witness and verify a counter activity. If verification confirms that the counter activity is ideal (see 'positive outcome' in Figure 9.1), the *Primary Activity* is executed. If verification confirms that the counter activity is sub-ideal (see 'negative outcome' in Figure 9.1), the counter actor executes *Pay Penalty* activity, which transfers an object *Penalty*, indicating a monetary penalty.

The process level solution of the Penalty pattern is variation of the Execution Monitoring pattern and can therefore be expressed by the same control principles. In addition, some pre-execution principles have been added. This is because the *Verify* activity must be executed before the *Pay Penalty* activity and the *Primary* activity must be executed after the *Pay Penalty* activity. These requirements can be relaxed if the primary actor trusts the counter actor to pay the penalty even if the *Primary* activity has been executed. The principles are listed in Appendix A.

9.1.4 Example

An example of this pattern is illustrated in Figure 9.2, in which we only focus on the value view since the process view is very similar to the Execution Monitoring pattern and has already been demonstrated in Chapter 8. In the ideal model, a primary actor *Buyer* and a counter actor *Seller* exchange value objects *Goods* and *Money*. In the sub-ideal model, a seller delivers damaged goods, which is modelled with the corresponding sub-ideal value object *Damaged Goods*. In return, a buyer does not pay, which is modelled with the sub-ideal value object *Money*.

Although both value transfers are sub-ideal, the actor behaving sub-ideally is the seller. This can be seen from the liability token **L** placed at the seller.

In the solution, the seller has to pay compensation for the damaged goods, which is modelled by a value object *Compensation* and corresponds to the value object *Penalty* of the Penalty pattern (see Figure 9.1).

If a penalty is to work, it should be sufficiently high for the counter actor. From the counter actor's *Seller* perspective, the value he accumulates as a result of delivering damaged goods and paying the penalty should be less than the value he accumulates when delivering non-damaged

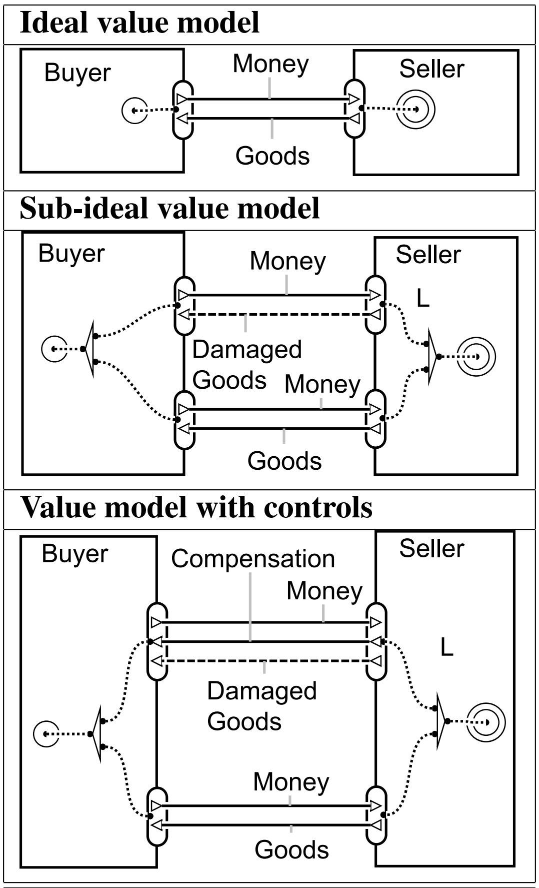


Figure 9.2: An example of the Penalty pattern

goods. From the primary actor's *Buyer* perspective, the compensation should compensate the loss he made because of the damaged goods delivered.

9.1.5 Related patterns

The process view of the penalty pattern is a variation of the Execution Monitoring pattern (see section 8.3).

9.2 Incentive pattern

A penalty is based on punishment and an incentive is based on rewards. To sum up, the incentive pattern rewards the counter actor for ideal behaviour by introducing a value object that increases the accumulated value of the counter actor in the *ideal path*.

9.2.1 Context

Value view. A primary actor and a counter actor execute a Primary and a Counter activity, as a result of which they exchange the value objects *PO* and *CO*.

Process view. At the process level, the value transfers *PO* and *CO* correspond to the execution of a primary activity and a counter activity correspondingly.

9.2.2 Problem

Value view. A counter actor may choose to behave ideally or sub-ideally, which corresponds to two dependency paths in the sub-ideal value model: the ideal path and sub-ideal path. The value transfers of the *ideal path* are the same as in the context model: *PO* and *CO*. In the *sub-ideal path*, the counter actor transfers a sub-ideal counter value object *Sub-ideal CO*, while the primary actor transfers the ideal value object *PO*. In addition, a liability token **L** is assigned to the counter actor to indicate that he is responsible for the sub-ideality.

Process view. In the process view, the transfer of the *Sub-ideal CO* corresponds to a sub-ideal counter activity. For brevity of representation, in the process model in Figure 9.1 we show the process that only corresponds to the sub-ideal path.

9.2.3 Solution

Value view. The *solution* is to prevent the sub-ideal value transfer *CO* by making a counter actor's value, accumulated in the ideal path, higher than a value accumulated in the sub-ideal

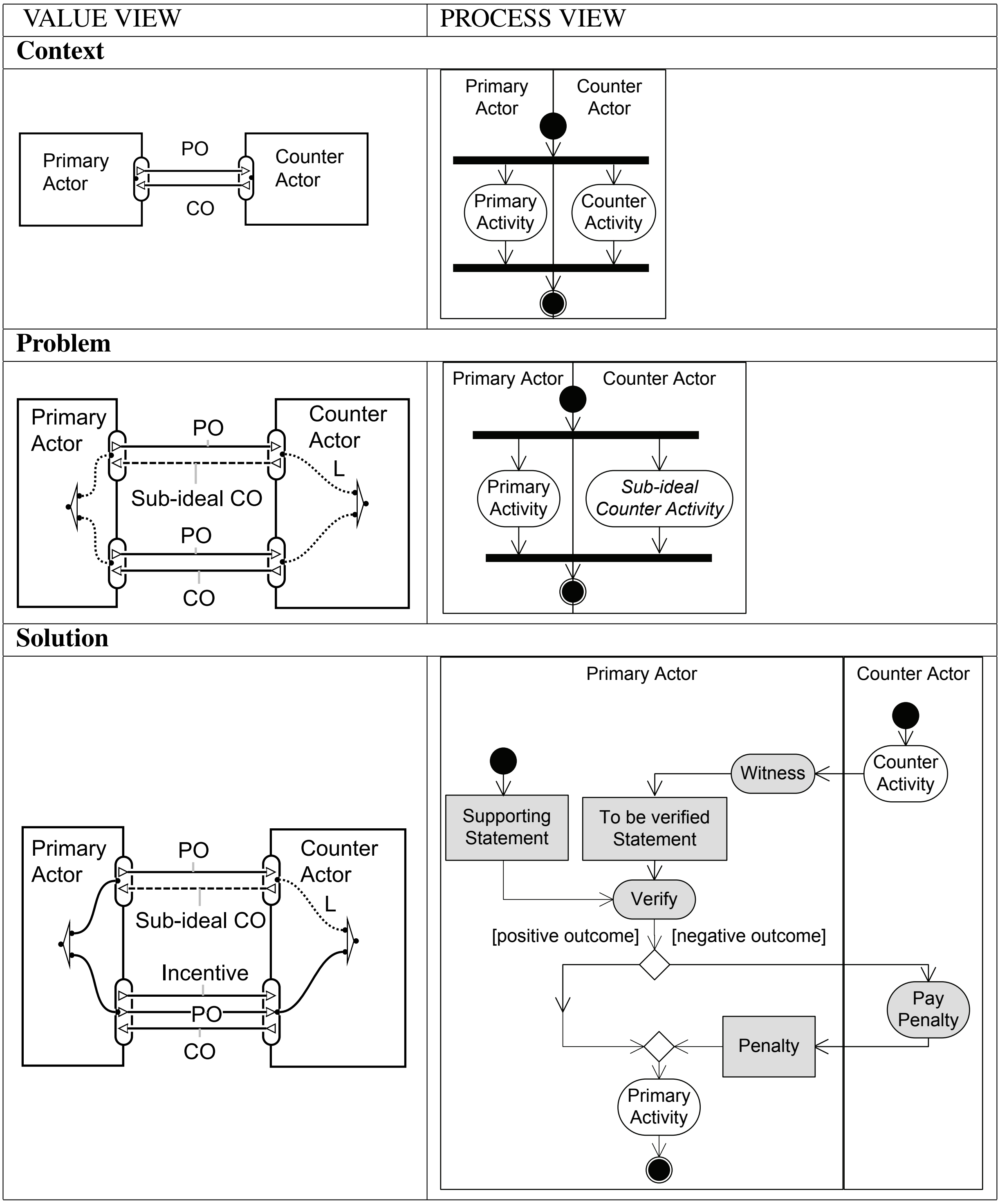


Figure 9.3: Incentive pattern

path. Such an incentive can be modelled in several ways. In Figure 9.3 the incentive is represented by a counter actor's incoming object *Incentive*, modelled in the same value interface as *PO* and *CO*. Other ways to model an incentive is as a subtraction from the *CO* (e.g. *CO - Incentive*) or as an addition to the *PO* (e.g. *PO + Incentive*), or as a separate value object *Incentive*, but in a different part of the ideal path. In general, the main requirements for the *Incentive* value object are that (1) it is added to the path that corresponds to the ideal behaviour of the counter actor, and (2) it increases the *accumulated value* of the counter actor. This last point will now be discussed in more detail.

Incentive size. BAs in the Penalty pattern, an incentive should make the value accumulated by the counter actor in the ideal path higher than the value accumulated in the sub-ideal path. In other words, the following should hold:

$$PO_{CA} - CO_{CA} + Incentive_{CA} > PO_{CA} - Sub - ideal\ CO_{CA}. \tag{9.4}$$

in which the left-hand side of the inequality represents the value accumulated by the counter actor in ideal path, and the right-hand side represents the value accumulated by the counter actor in the sub-ideal path. With the lower index CA we indicate here that the objects *PO*, *CO* and *Sub-ideal CO* are valued from the subjective perspective of the *C*ounter *A*ctor. They are non-monetary, whereas the *Incentive* is monetary.

Taken from his perspective, a primary actor should not create incentives which lead to losses in value. An incentive should therefore not make the value accumulated by the primary actor in the ideal path any less than the value accumulated in the sub-ideal path:

$$CO_{PA} - PO_{PA} - Incentive_{PA} \geq Sub - ideal\ CO_{PA} - PO_{PA}, \tag{9.5}$$

in which the left-hand side of the inequality represents the value accumulated by the primary actor in the ideal path, and the right-hand side represents the value accumulated by the primary actor in the sub-ideal path. With the lower index PA we indicate that the objects *PO*, *CO* and *Sub-ideal CO* are valued from the *P*rimary *A*ctor's subjective perspective. They are non-monetary, whereas the *Incentive* is monetary.

Process view. At the process level, the solution requires verification that a counter activity has been performed ideally before an incentive is handed over (see Figure 9.3). This solution is a variation of the solution given by the Execution Monitoring pattern, which was discussed in detail in section 8.3. In short, the primary actor has to witness and verify the counter activity. If verification confirms that the counter activity is ideal (see 'positive outcome' in Figure 9.3), the *Pay Incentive* activity and the ideal *Primary* activity are executed. The *Pay Incentive* activity denotes a collection of operational activities which are required to transfer the object *Incentive*. If the verification finds the counter activity sub-ideal (see 'negative outcome'), the incentive is not paid.

Control in an Incentive pattern is less strict than in a the Penalty pattern, in the sense that it does not prohibit execution of a primary activity in a sub-ideal situation. Unlike the Penalty pattern, a primary activity is always executed, even in return for a sub-ideal counter activity.

As in the Penalty pattern, the control mechanism at the process level can be expressed with the same control principles as for the Execution Monitoring pattern. In addition, some conditions have been added which relate to the transfer of an incentive. This is because the solution requires the *Pay Incentive* activity to be executed before the *Primary* activity and after the *Verify* activity.

9.2.4 Example

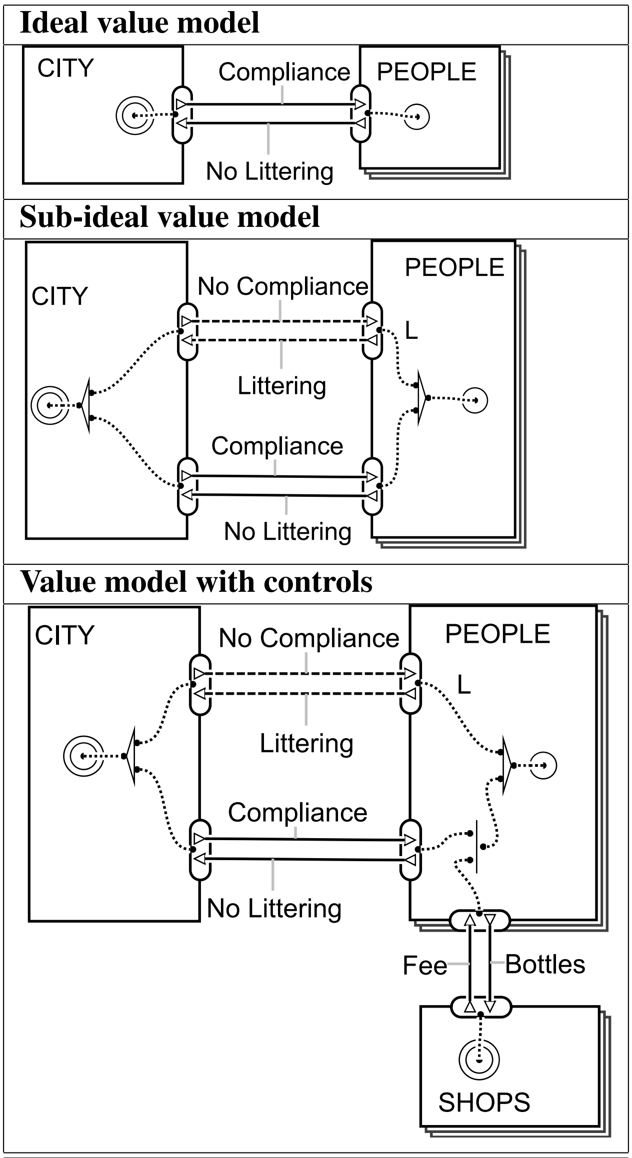


Figure 9.4: An example of an Incentive pattern

An example of an Incentive pattern is illustrated in Figure 9.4. It represents a value transfer between a city council, say *City*, and people who live there, say *People*. Taken from the city council's perspective, the ideal behaviour of people is that they do not throw garbage on the streets, i.e. do not litter. In the ideal situation we therefore model the counter actor *People* to deliver a value object *No Littering* to the city. In return the city states that a person complies to existing laws about not littering, which is modelled with the value object *Compliance* exchanged in return for *No Littering*. In fact, a person always has compliance except when caught littering.

In the sub-ideal model, the sub-ideal path represents the behaviour of those people who litter. In most cases littering is not detected since the police and other government agencies cannot catch every person who litters. Thus, the sub-ideal counter value object *Littering* is transferred in return for the ideal primary value object *Compliance*.

The city council can introduce incentives against littering by rewarding good behaviour. For example, in the Netherlands, there is a system called 'Statiegeld', which motivates people to return empty bottles to a shop by giving them an amount of money in exchange for them. In Figure 9.4 we model a shop that has a bottle recycling point; people bring the *Bottles* to the shop and get a *Fee* back. As in the case of the Penalty pattern, such an incentive mechanism also requires witnessing. However, it requires witnessing of the ideal behaviour (bottle recycling) rather than of the sub-ideal behaviour (littering).

This example demonstrates an incentive *Fee* which is not part of the ideal value transfers *No Litter* and *Compliance*, as in Figure 9.3, but a part of another value transfer in the ideal path of the counter actor *People*.

From the perspective of the counter actor *People*, the value of returning the bottles should be higher than the value of throwing them on the streets. This value is subjective for every person. If we assume that people assign no value to empty bottles, then they will always prefer to return them and will not litter. However, for people who collect bottles, who are lazy, or who live far from the bottle recycling point, taking the bottles to a recycling point will create costs and thus reduce the accumulated value. These people may not be motivated by the incentive and will proceed with littering. From the perspective of the primary actor *City* running the bottle recycling point should be less expensive than collecting bottles from the streets.

9.2.5 Related patterns

The process view of the Incentive pattern is a variation of the Execution Monitoring pattern (see section 8.3).

9.3 Summary

In this section we have introduced two contractual patterns: Incentive and Penalty. The Penalty pattern describes a mechanism of punishing sub-ideal behaviour and the Incentive pattern describes a mechanism of rewarding ideal behaviour. In these patterns the control mechanism affects both the value and the process viewpoint. To model a penalty we use the value object *Penalty*, which is added to the sub-ideal path. To model an incentive we use the value object *Incentive*, which is added to the ideal path.

In addition, we have discussed issues related to determining the size of a monetary penalty and incentive. As was argued, we need to know the value of other objects in the ideal and sub-ideal paths in order to define the size or amount of a penalty or incentive. In general, penalties and incentives should make sure that the accumulated value of a counter actor who behaves sub-ideally is less than the accumulated value for ideal behaviour. The primary actor should not make losses by paying incentives, and he should be compensated by the penalty.

At the process level, the Incentive and Penalty patterns represent variations of the Execution Monitoring pattern. In the Penalty pattern, a primary actor should verify a counter activity and demand a penalty only if the activity is sub-ideal. In the Incentive pattern, a primary actor only pays an incentive if the counter activity is ideal.

Chapter 10

Delegation patterns

So far now we have described six control patterns, in which we assume a value transfer takes place between two actors: a primary actor and a counter actor. In reality, the two actors often delegate activities to other actors. It can be necessary for both control and business reasons. A control-related reason for delegation is segregation of duties: sometimes the procedural pattern prescribes that an activity has to be delegated to avoid violation of the segregation of duties. This was demonstrated in the health care case study in Chapter 5, in which the needs assessment and product allocation functions were assigned to different actors to avoid a conflict of interests. Segregation of duties may be required in the control patterns as stated in the Activities-Actors principles, see Chapter 7. An example of a business-related motivation for delegation is when it is more efficient to delegate an activity to another actor rather than to execute it yourself.

Delegation should not result in violation of the control principles described in the original control pattern. In **delegation patterns** we describe how delegation has to be performed to ensure that all the control principles are observed and that the delegated activity is executed ideally.

Delegation patterns have a *context-problem-solution* structure. A problem in a delegation pattern is related to a sub-ideal execution of a control activity which can emerge because of its delegation. The solution given by a delegation pattern describes how a control pattern has to be adjusted to prevent this control problem.

In this the chapter we present two delegation patterns: Simple Delegation pattern and Testifying Chain pattern. In the Simple Delegation pattern we consider an initial situation (the context) in which the output of a control activity *Statement* is received by the same actor who executes the control activity. In the Testifying Chain pattern we consider an initial situation (the context) in which the output of a control activity is received by an actor who is not the one who executes the control activity. We consider delegation of the control activity in both patterns.

10.1 Delegation Box

To illustrate the delegation more precisely, we introduce a **delegation box** which is drawn in Figure 10.1 with a dashed square. The box surrounds the *unit of delegation*, which is an activity or a part of a process that is delegated. In Figure 10.1 a unit of delegation is an *Activity A*. The box is connected to inputs and outputs of the unit of delegation. An **incoming point** is depicted with a white circle, and it indicates a point of connection of the incoming objects of the unit. An **outgoing point** is depicted with a black circle, and it indicates the point of connection of the outgoing objects of the unit.

In Figure 10.1, (a) the *Activity A* and the delegation box are placed between two swim-lines, which indicate an assignment of *Activity A* to *Actor A*. The fact that the incoming and outgoing points are placed within *Actor A* indicates that this actor receives inputs and outputs of *Activity A*.

The inputs and outputs of *Activity A* are modelled with objects *Incoming Object* and *Outgoing Object* correspondingly. The *Incoming Object* is always connected to the incoming point of the delegation box and the *Outgoing Object* is always connected to the outgoing point of the delegation box.

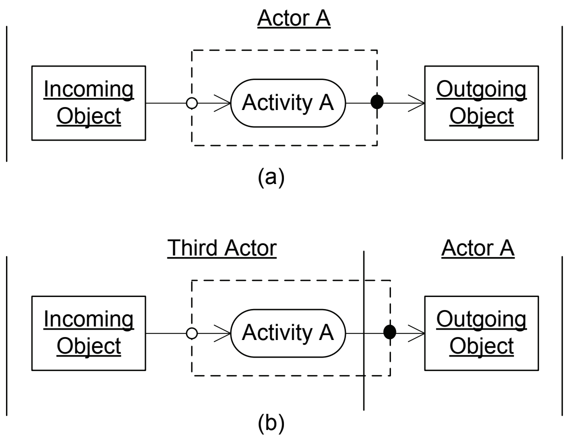


Figure 10.1: The Delegation Box

In Figure 10.1, (b) we model a situation in which *Actor A* delegates *Activity A* to a *Third Actor*. Certain rules should be observed in such a situation. Firstly, the *Incoming Object* should be received by the actor who executes the delegated activity. Secondly, the *Outgoing Object* should be received by the actor who delegates *Activity A*.

The following rules can be defined to ensure that the objects move correctly, within the terms of the delegation box:

- The incoming point of the delegation box moves to the actor who executes the delegated activity.
- The outgoing point of the delegation box stays at the actor who delegates the activity.
- The Incoming Object and Outgoing Object should remain connected to the incoming and outgoing points of the delegation box correspondingly;

Following these rules always ensures that the incoming and outgoing objects remain in the correct position, that they are not missing and that they are received by the correct actor after the activity is assigned to another actor.

10.2 Simple Delegation pattern

The process-level solutions of procedural and contractual patterns require the addition of control activities to either a primary or a counter actor. Control activities include witness, verify, testify, and promise activities (see Chapter 7). However, in some cases, the primary and counter actors are not able to execute these activities, as already explained, in which case these activities should be executed by third parties on behalf of the primary or counter actor. Such a delegation may cause sub-ideal execution of the control activity. The Simple Delegation pattern, as shown in Figure 10.2, describes such a situation and suggests a solution to avoid it.

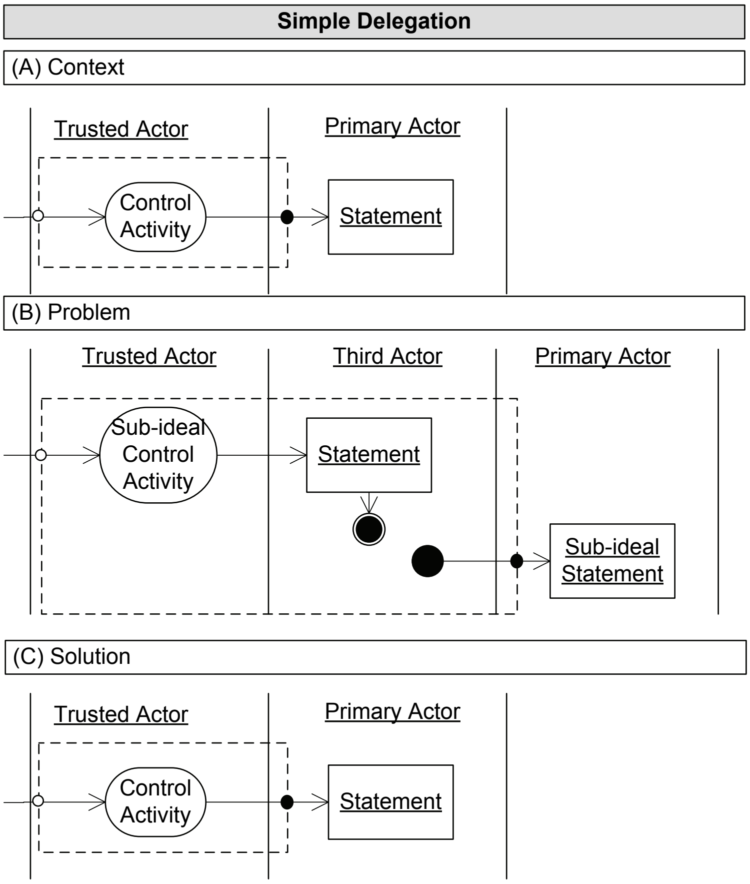


Figure 10.2: Simple Delegation pattern

10.2.1 Context

In the context we consider that a primary actor delegates the *Control Activity* to a *Trusted Actor*. Thus, the primary actor is a beneficiary of the control activity, while the execution is done by a trusted actor. The outcome of the *Control Activity* is an object *Statement* and it is received by the primary actor, as a beneficiary of the control activity.

The model of the context is created by moving the control activity to the third actor from another 'no-delegation' model where the control activity is executed by a primary actor. To ensure the situation with delegation has proper inputs and outputs, we followed the rules stated in Figure 10.1. We moved the incoming point of the delegation box to the third actor and left the outgoing point at the primary actor.

10.2.2 Problem

Such a delegation can lead to several problems, as a result of which the *Control Activity* can be executed sub-ideally. Other possible problems are that the *Trusted Actor* is not independent and socially detached from the counter actor and that the *Statement* is transferred from the *Control Activity* to the primary actor *indirectly*.

These problems are represented in Figure 10.2 by depicting a Sub-ideal Control Activity instead of an (Ideal) Control Activity. In addition, the link between the control activity and the statement received by the primary actor is broken. Finally, the indirect transfer is modelled by showing that the statement is transferred to a primary actor via a third actor.

10.2.3 Solution

To ensure that the *Control Activity* is executed ideally, the primary actor has to observe the following rules:

- The *Trusted Actor*, who executes *Control Activity*, should be independent and socially detached from the counter actor.
- The *Statement* should be transferred *directly* from the *Control Activity* to the primary actor.

10.2.4 Example: Certification

This Simple Delegation pattern can be combined with the process-level solution of any of the six patterns described in Chapter 8 and Chapter 9. We illustrate the application of this pattern in combination with the pattern Partner Screening. We call this particular combination **Certification** and, though it is not the only possible combination, it was observed most frequently in the case studies.

Figure 10.3 shows the solution of the Partner Screening pattern, which requires the primary actor to execute *Witness*, *Verify* and *Promise* control activities. The delegation box indicates which activities are to be delegated to a third actor. As we can see, the *Witness* and *Verify* activities are to be delegated as well as the generation of the *Supporting Document*. In Figure 10.4 and 10.5 we show changes in the process and value models of the solution after the Simple Delegation pattern is applied.

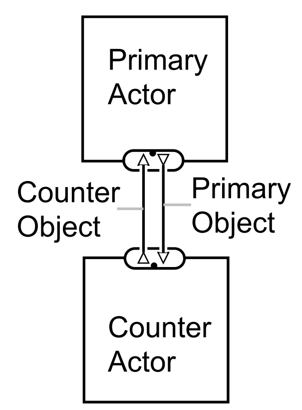
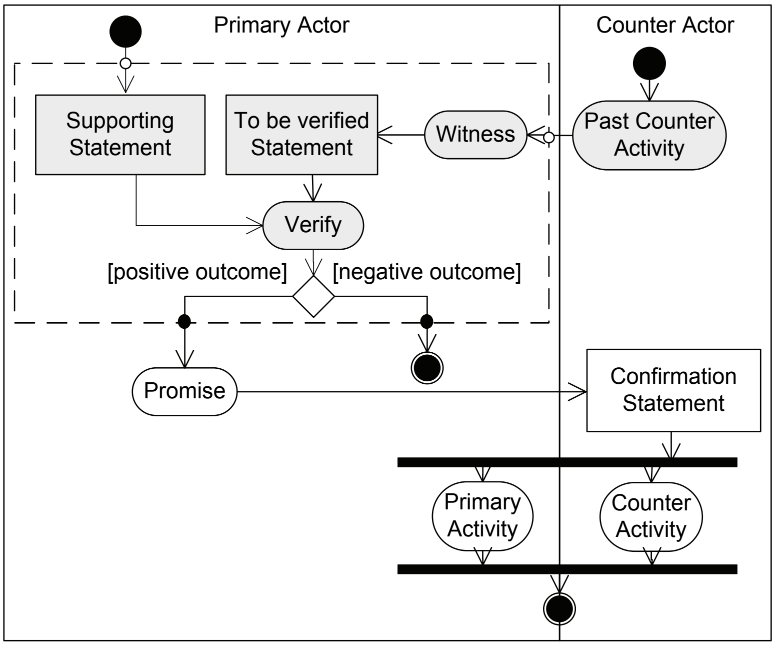
Solution: Value view	Solution: Process view
	

Figure 10.3: Solution of the Partner Screening pattern

Process view. As in the Partner Screening pattern, the primary actor in Figure 10.4 must witness the past activities of the counter actor and verify compliance with certain standards the primary actor maintains. Suppose, the primary actor has to delegate the verification and witnessing activities to a trusted actor, e.g. because it is more efficient. The trusted actor verifies the past activities of the counter actor and, if the outcome of the verification activity is positive, it generates a *Positive Statement*, which is transferred directly to the primary actor. The primary actor must receive a *Positive Statement* from the trusted actor before executing the *Promise* activity and sending *Confirmation Statement* to the counter actor. Should verification produce a negative outcome, the primary actor does not receive a *Positive Statement*, but is notified by the Trusted Actor about the negative outcome.

Value view. The effect of delegation also changes the value view, unlike in the Partner Screening pattern. The result is shown in Figure 10.5. Firstly, the trusted actor is added to the value model. Secondly, the value transfers between the trusted actor and the primary actor are added. At the process level, the trusted actor executes *Witness* and *Verify* activities on behalf of the primary actor. At the value level this corresponds to a value transfer from the trusted actor to the primary actor. We call this value transfer *Partner Screening (Witness&Verfy)*. In return the primary actor transfers *Payment*, because we assume that this service is provided for money.

In addition, the primary actor has an OR fork, which models a positive outcome of the partner screening service, in which case, the primary actor exchanges values with the counter actor. Otherwise, the exchange does not take place and the OR fork leads to the boundary element.

Note that in the process model we omit the processes that correspond to the transfer of the value object *Payment*. The process model must be revised one more time to add these corresponding

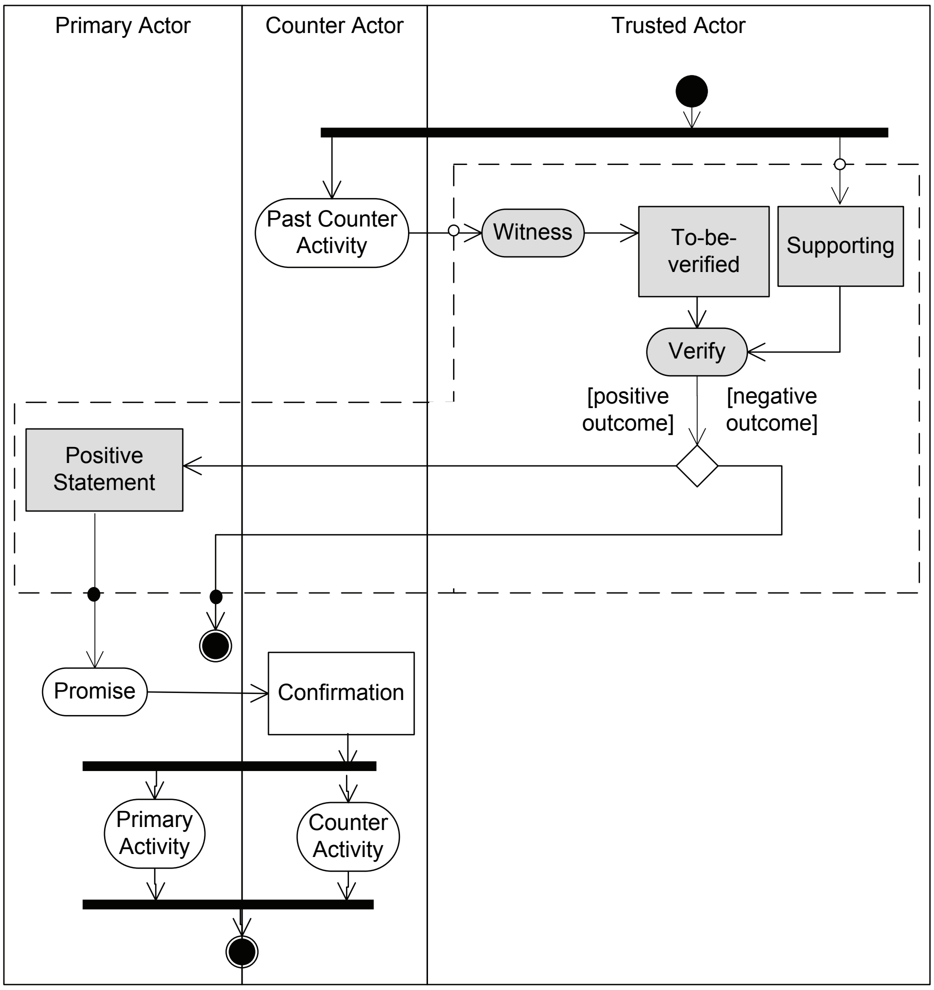


Figure 10.4: Certification, process view: the Partner Screening pattern combined with the Simple Delegation pattern

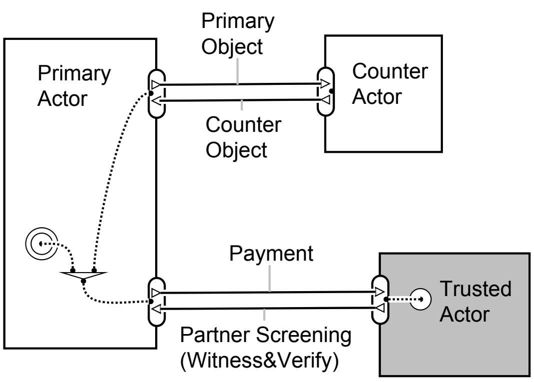


Figure 10.5: Certification, value view: the Partner Screening pattern combined with the Simple Delegation pattern

processes.

Example of Certification: health care case revisited

Let us reconsider the health care case study in Chapter 5. In this case study, the administrative office can only make contracts with the care providers that are accredited by the government agency CVZ. The accreditation by CVZ is an example of Certification. The process model of the example is shown in Figure 10.6.

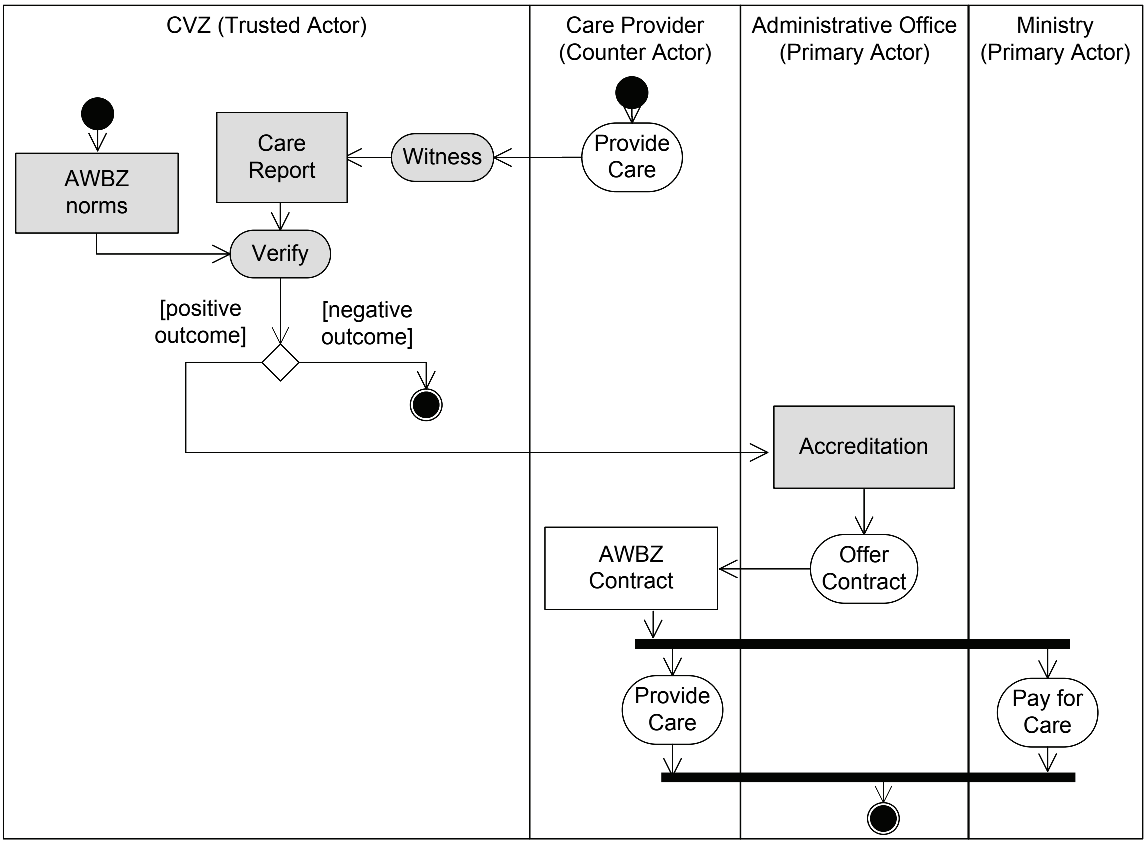


Figure 10.6: Certification, an example in healthcare

The *ministry of Health* contracts *care providers* to deliver AWBZ services. The ministry wishes to ensure that only those care providers who are really able to deliver the services, get the contract. In this case, the ministry is the primary actor and the care providers are counter actors. According to the Partner Screening pattern, the ministry has to screen every care provider and only contract one if the check reveals the appropriate abilities to provide care. Obviously, although the ministry is responsible for all these tasks, the actual execution is delegated to government agencies. To be specific, the contracting process is performed by the *administrative office* and the screening of care providers is performed by *CVZ*. Within the scope of the Certification model shown in Figure 10.4, the CVZ plays a role of the trusted party.

As the process model in Figure 10.6 shows, CVZ first verifies that a specific care provider is capable of providing AWBZ care and, if this is the case, then informs the administrative office by providing an accreditation for the care provider. The *Accreditation* object is an instance of a *Positive Statement* in the Certification model in Figure 10.4. Furthermore, the contract with the care provider as well as the delivery of services by the care provider (see the counter activity *Provide Care*) are only executed after accreditation has been issued.

This example is more complex than the Certification model described in Figures 10.4 and 10.5. The role of the primary actor is played by two parties: the ministry and the administrative office. The contracting process (the promise activity *Offer Contract*) is performed by the administrative office and the primary activity *Pay for Care* is performed by the ministry.

10.3 Testifying Chain pattern

The Testifying Chain pattern considers a situation in which the beneficiary of the control activity is some actor, called a third actor. Let us consider that he delegates execution of the control activity to a primary actor. Furthermore, the primary actor delegates the execution to the trusted actor. As a result, the responsibility of the execution lies on a primary actor, while the actual execution is done by a trusted actor. The situation is more complicated with the fact that the third actor only trusts the primary actor and not the trusted actor. Thus, the primary actor should guarantee reliable execution of the control activity to the third actor.

10.3.1 Context

In the context in Figure 10.7 the primary actor delegates the *Control Activity* to his *Trusted Actor*. Thus, the *Control Activity* is executed by a trusted actor. The outcome of the activity, *Statement*, is transferred to a third actor. The primary actor has no role in the process.

10.3.2 Problem

The third actor trusts the primary actor, but not his trusted actor. Thus, from the third actor's perspective, the control activity may be executed sub-ideally, which is modelled in the problem section in Figure 10.7. Because the primary actor has responsibility for the control activity, the third actor will accuse the primary actor in case of a sub-ideal execution of the *Control Activity*. Thus, some measures should be taken against the sub-ideal execution by the trusted actor.

10.3.3 Solution A: Low trust

To ensure that the control activity is executed in a proper way, the *Statement* should be testified to by the primary actor before being sent to the third actor. This is needed because the third actor will trust the control activity only if it is executed, or at least testified, by the primary actor.

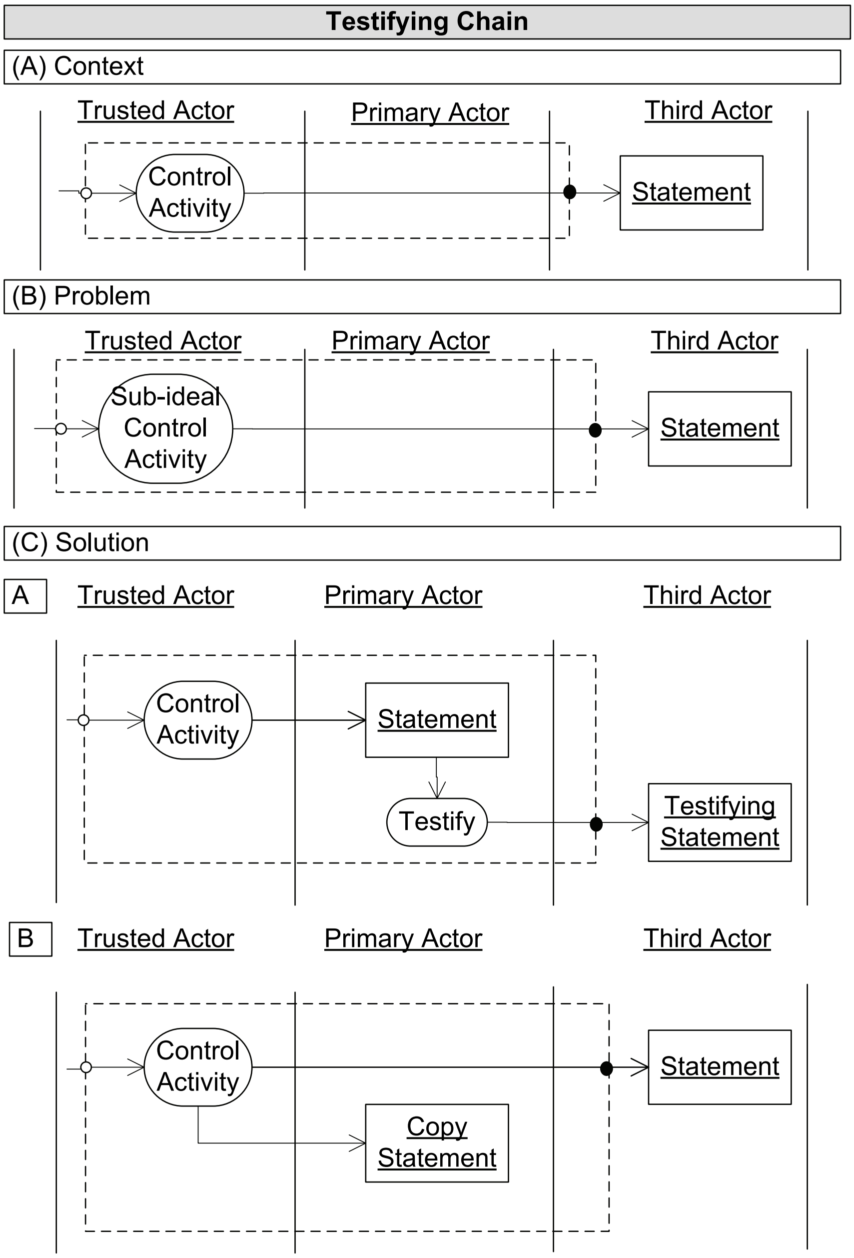


Figure 10.7: Testifying Chain pattern

Solution *A* in Figure 10.7 (C) shows that the trusted actor sends the *Statement* to the primary actor, who testifies to it with a *Testify* activity and then sends it on to the third actor. The statement received by the third actor is a *Testifying Statement*.

Delegation can be repeated several times to form a *chain of testifying statements*. The trusted actor can delegate *Control Activity* to his (second) trusted actor, who, in turn can then delegate it to a third trusted actor and so on. This chain of statements goes back to the *audit trail* concept in the internal control theory (see section 2.1.1). With help of a chain of statements we can trace the actor who executes the *Control Activity* and the one who testifies its execution.

Some control principles must be observed here. According to the Testified - Testify Principles the *Testify* activity must be performed *after* the testified *Control Activity*. In addition, the *Testifying Statement* has to be transferred directly to the third actor and the *Statement* has to be transferred directly to *Testify* activity.

10.3.4 Solution B: High trust

In Solution A we state that the primary actor must testify to the outcome of the *Statement*, which, on the other hand, creates an extra administrative burden. The *Statement* has to be processed by the primary actor, and this can create delays, which is not the case in a situation in which the statement is sent directly from the trusted actor to the third actor.

If the primary actor is really sure that the trusted actor executes the *Control Activity* ideally, the primary actor can entitle the trusted actor to send the *Statement* directly to the third actor without extra testification. In this case, a proper legal entitlement of the trusted actor by the primary actor is needed, allowing that the trusted actor could act *on behalf* of the primary actor. To remain in control, the primary actor has to have a copy of the *Statement*.

This kind of delegation is described Solution B in Figure10.7. The *Control Activity* is executed by the trusted actor, the *Testify* activity is not added and the trusted actor sends the *Statement* to the third actor and sends a *Copy Statement* to the primary actor.

10.3.5 Example

An example of this pattern will be demonstrated in the beer living lab case study in Chapter 13.

10.4 Summary

In the control patterns in Chapters 8 and 9 we assumed exchanges between two actors: a primary actor and a counter actor. In reality, the two actors often delegate activities to other actors, which may result in control problems and weaken the control mechanism as suggested by the control pattern. In this chapter we introduce **delegation patterns** to describe how delegation has to be executed to prevent the sub-ideal execution of the control activity.

The delegation patterns describe control problems caused by delegation and the solution to this problem. From our examination of two kinds of delegation, we have determined two patterns:

Simple Delegation pattern and Testifying Chain pattern. The first pattern considers the delegation of a control activity by a primary actor when the outcome of the control activity is also received by the primary actor. The second pattern considers the same situation, but when the outcome of the control activity is received by an actor other than the primary actor. The control problem in both cases is that, when a control activity is delegated, it is executed sub-ideally. The patterns identify control problems for each situation and suggest a solution.

Chapter 11

The e^3*control* design framework for patterns

In this section we describe the steps to be performed in designing controls with the e^3*control* patterns. The process consists of five steps, which are an extension of the three steps of e^3*control* in Chapter 3 adjusted to incorporate the control patterns in the design process.

The steps and their outcomes are schematically shown in Figure 11.1. As in e^3*control*, the design process starts with the design of an ideal situation in Step 1 and a sub-ideal situation in Step 2. In Step 3 the user selects an appropriate pattern to mitigate the control problem identified in the Step 2. In Step 4 the user designs a solution to a problem by applying the selected pattern.

Steps 1 to 4 can be repeated to deal with various control problems. For example, if several control problems have been identified, Steps 1 to 4 have to be applied for each of them separately. The single execution of the four steps is referred to henceforth as an ***e*^{3}*control* cycle**.

Each step results in design artifacts, e.g. e^3*control* models or process models, which we refer to as **outcomes** of the step. The outcome of Step 1 is an ideal value and an ideal process model. The outcome of Step 2 is a sub-ideal value and a sub-ideal process model. The outcome of Step 4 is a solution value and a solution process model.

In addition to the description of each step, we discuss an Example, Guidelines, and References. In the Example we illustrate the step using an educational example. In Guidelines we describe how to handle more complex situations that can occur in this step. In References we provide a reference to the theory (described in this thesis) that should be familiar to the user when performing the step.

11.1 Step 1: Design an ideal situation

The first step considers the design of an ideal situation. As defined in Chapter 3, an ideal situation is a situation in which counter parties are supposed to behave without errors or irregularities.

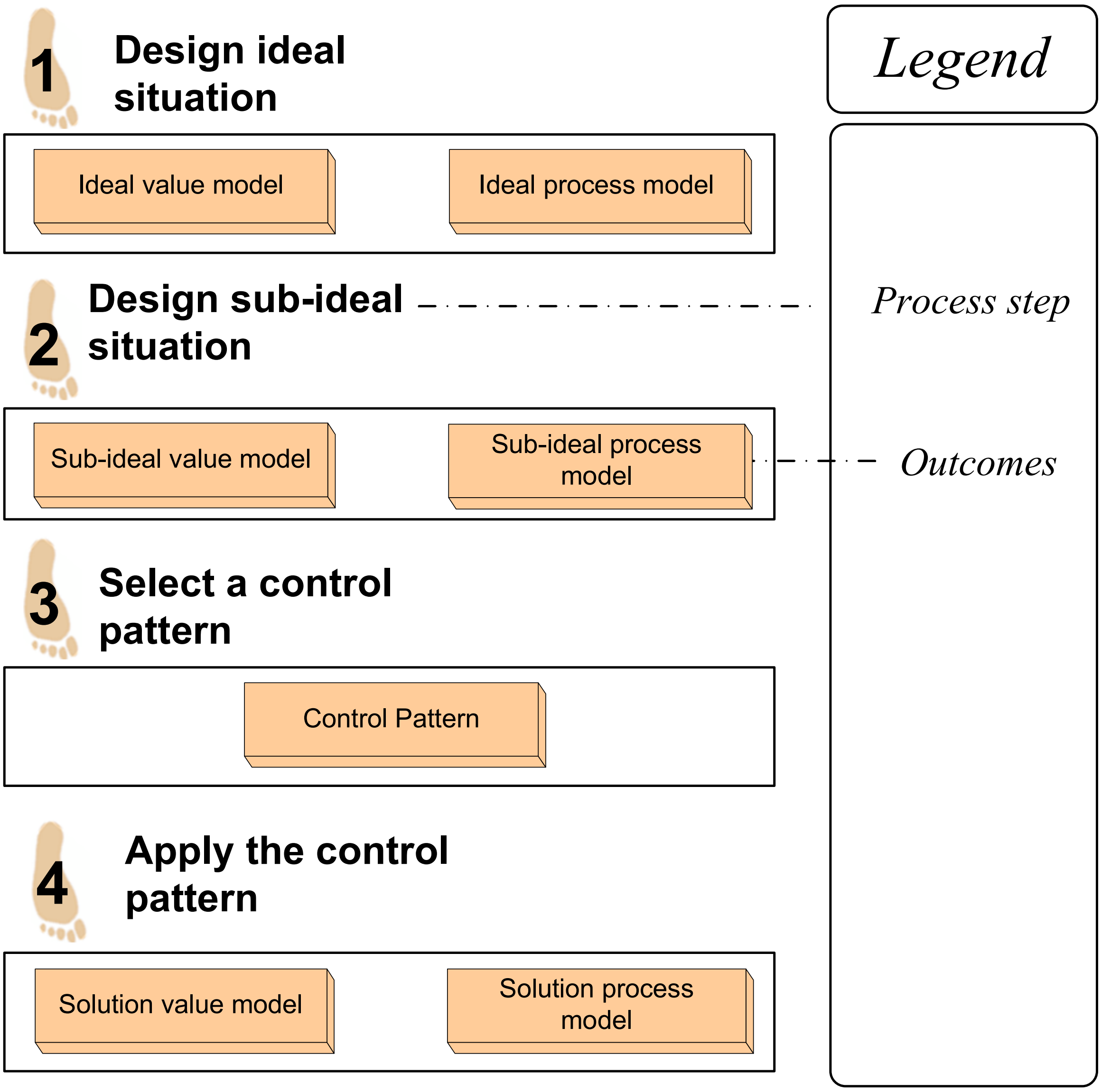


Figure 11.1: Steps to design controls with control patterns in *e³control*

An ideal value and an ideal process model are developed in this step. A value model is required to analyze and design value-related aspects of controls, and a process model is necessary to analyze and design procedural aspects of controls. An ideal value model should be developed using the *e^3value* ontology (see section 3.1) and an ideal process model should be developed using a process modelling technique. In our case, we use UML activity diagrams [Fowler and Scott, 2000].

11.1.1 Guidelines

Guideline 1: Controls in an ideal model. Notice that an ideal model can still include some controls and, in practice, some elements of controls are already present in an ideal model. An ideal situation is defined as a situation with no errors or irregularities, and with ideally behaving actors, which does not imply that there are no controls included in the ideal model. If these controls are efficient, then risks will not materialize. Controls that are inefficient will not mitigate the risks, so the old controls have to be improved or new controls have to be introduced.

11.1.2 Example

In the educational example we consider three actors: a buyer, a seller, and a carrier. The ideal process and value models developed in Step 1 are shown in Figure 11.2.

A buyer buys some goods from a seller in return for a *Goods Fee*. In addition, the goods are delivered to the buyer by a carrier, who receives the *Delivery Fee*. As can be seen from the process model, the buyer makes the payment after the goods have been shipped and delivered.

These models are usually built based on data collected by a user and with the assumption that all actors behave ideally. A definition of what is ideal or what not is also given by the user according to the points of view taken by the stakeholders in a case study.

11.1.3 References

Ideal situations and ideal value models are discussed in Chapter 3, section 3.1. For more references on the *e^3value* methodology for building ideal value models see [Gordijn, 2002].

11.2 Step 2: Design a sub-ideal situation

The second step considers the design of a **sub-ideal situation**. As defined in Chapter 3, a sub-ideal situation is a situation in which network participants behave with errors or irregularities, and this leads to **control problems**.

In this step, sub-ideal value and sub-ideal process models should be developed to analyze value-related and procedural aspects of a control problem. The sub-ideal value model should be developed using the *e^3control* ontology (see Chapter 3). The sub-ideal process model should be developed using a process modelling technique such as a UML activities diagram.

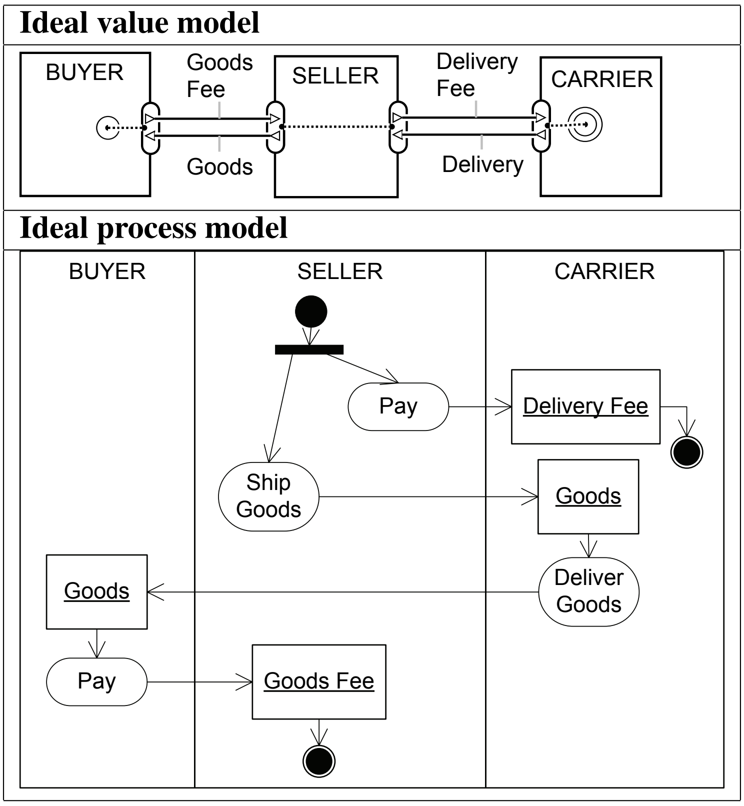


Figure 11.2: An example of Step1

11.2.1 Guidelines

Guideline 2: Multiple control problems. Multiple control problems can occur when, for example, multiple actors behave sub-ideally or when one actor causes several problems. It is more convenient to consider **one control problem** in one *e^{3}control* cycle at a time. If multiple control problems are detected during this step, just one of them should be selected to proceed and the rest should be tackled by repeating Steps 1 to 4 in the next *e^{3}control* cycle.

Note that in the methodology we do not describe considerations behind implementing controls in a particular order, even though this choice can influence the final result, which is known as *feature interaction* in computer science. This is an important issue and is a subject of future research.

Guideline 3: Using patterns to identify control problems. Patterns can be used not only to design control mechanisms, but also to identify control problems. The user of the methodology may collect data while being familiar with certain patterns and being aware of the control problems described by them. Such a design process is accepted in the patterns literature [Gamma et al., 1995]. However one should not forget that the *e^{3}control* patterns do not describe *all* possible controls problems. For example, control problems related to information system-level of analysis are not covered by the *e^{3}control* patterns.

11.2.2 Example

There are multiple potential control problems in our educational example. Suppose, we identify one important control problem during the investigation of our case and we consider it further. The risk taken by the seller is that the buyer may claim that goods have not been received and, based on this claim, he refuses to pay for the goods. A similar problem is when the buyer claims that he has not received the goods that he ordered and demands compensation or change of the goods.

The control problem is modelled with sub-ideal value and process models in Figure11.3. The sub-ideal value model shows a sub-ideal value transfer of the object *Goods Fee*. The corresponding sub-ideal value object is called *No Goods Fee*. The model also specifies that the reason for the missing value object is the claim of non-delivery by the buyer. In the sub-ideal process model the activity *Pay* of the buyer is not executed; instead, the buyer executes the activity *Claim Wrong/No Delivery*, which results in the delivery of a *Claim* to the Seller.

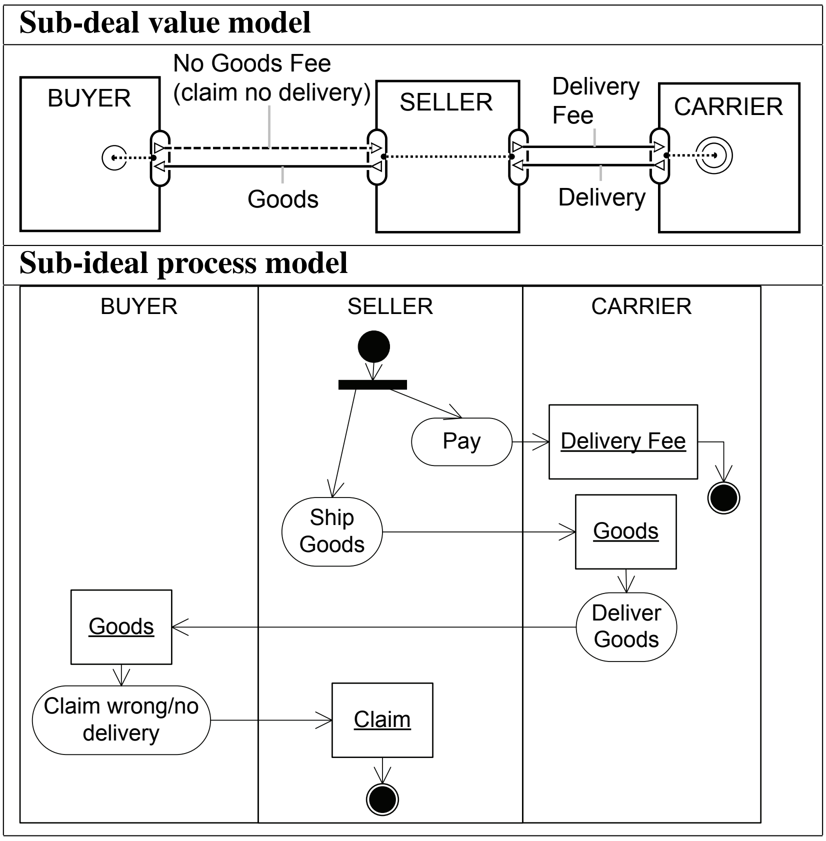


Figure 11.3: An example of Step 2

11.2.3 References

The sub-ideal situation and the sub-ideal value model are discussed in Chapter 3.

11.3 Step 3: Select a control pattern

In this step we need to select a control pattern which suggests an appropriate mechanism against the control problem identified in the Step 2. We start by identifying the primary and counter actor and the primary and counter activities and value objects. Then, we identify the pattern to be applied to mitigate the selected sub-ideal counter activity according to type.

11.3.1 Step 3.1. Identify ideal and sub-ideal primary and counter actors, value objects, and activities

In this step we identify which actors play the roles of the primary actor and the counter actor. We also identify primary and counter value objects as well as primary and counter operational activities for both the ideal and sub-ideal situation. We now go on to describe each element that should be identified.

Primary Actor. A pattern always considers one primary actor. The sub-ideal value model from Step 2 should be used to find a primary actor in a case. Use the following guidelines:

1. A primary actor does not trust the counter actor (see below) with respect to the execution of some value transfer and expects him to behave sub-ideally;
2. The sub-ideal behaviour is defined subjectively by the primary actor;
3. A primary actor plays the role of a *principal*.

Counter Actor. A pattern always considers one counter actor. The sub-ideal value model developed in Step 2 should be used to find a counter actor in a case. Use the following guidelines:

1. A counter actor behaves sub-ideally and causes *sub-ideal value transfers*;
2. In graphical models a counter actor has a *liability token*;
3. A counter actor plays the role of an *agent*.

Sub-ideal Counter Value Object. A sub-ideal counter value object is a result of errors or irregularities made by the counter actor. The sub-ideal value model should be used to find a value object that corresponds to the control problem described in Step 2. This is a sub-ideal value object. The value transfer that transfers this value object is marked with a dashed line.

Ideal Counter Value Object. The ideal value model should be used to find a value object that corresponds to the sub-ideal counter value object. This is an ideal counter value object

Ideal Primary Value Object. A primary value object is exchanged *in return* for a counter value object. The ideal value model should be used to find the primary value object.

Sub-ideal Primary Value Object. In some cases the transfer of the sub-ideal counter objects causes sub-ideality of the primary value object. The sub-ideal value model should be used to find a sub-ideal value object that corresponds to the primary value object. This is a sub-ideal primary value object.

Ideal Primary Activities. The ideal process model should be used to identify an activity or a set of activities that correspond to the value transfer of the primary value object. The primary activities are executed by the primary actor or delegated by the primary actor to other actors.

Sub-ideal Primary Activities. If the sub-ideal primary value object exists, there will also be sub-ideal primary activities. The sub-ideal process model should be used to identify an activity or a set of activities that correspond to the value transfer of the sub-ideal primary value object.

Ideal Counter Activities. The ideal process model should be to identify an activity or a set of activities that correspond to the value transfer of the counter value object. The counter activities are executed by the counter actor or delegated by the counter actor to other actors.

Sub-ideal counter activities. The *Sub-ideal Counter* activities can be found in the sub-ideal process model. They are sub-ideal versions of the ideal counter activities. As already identified in section 7.3, the following sub-ideal counter activities are possible:

- A counter activity executed at a wrong time or in the wrong sequence
- A counter activity not executed at all
- A counter activity executed without proper authorization
- A counter activity executed by a wrong actor
- An activity involving a wrong resource
- An activity involving a wrong amount or number of resources
- An activity executed at the wrong location

In addition to sub-ideally executed counter activities, the sub-ideal process model may contain activities known as *Emerging Sub-ideal* activities that do not exist in the ideal value model. For example, forging a document necessitates the execution of sub-ideal activities, e.g. printing the forged document. In particular, we distinguish two types of such sub-ideal activities: *Deny Commitment* and *Deny Execution*.

To summarize, we distinguish *four types of sub-ideal activities*:

1. **Sub-ideal Counter** activity is an ideal counter activity executed sub-ideally;
2. **Deny Commitment** activity is an Emerging Sub-ideal activity by which the counter actor claims that he made no obligations to transfer the CO to the primary actor;
3. **Deny Execution** activity is an Emerging Sub-ideal activity by which the counter actor claims that the primary actor executed the primary activity sub-ideally.

4. **Other Emerging Sub-ideal** activities are activities, which appear as a result of sub-ideal behaviour;

11.3.2 Step 3.2. Match the sub-ideal activity to a pattern

A sub-ideal activity has now been selected which can be one of four types: Sub-ideal Counter, Emerging Sub-ideal, Deny Commitment and Deny Execution. If the selected sub-ideal activity matches one of these types, a control pattern can be selected in two steps.

Select the pattern according to the sub-ideality type. Activities of the *Sub-ideal Counter* and/or *Emerging Sub-ideal* types signal control problems described in the Partner Screening pattern, and Execution Monitoring and its contractual variations Penalty and Incentive patterns. Therefore, if an activity of one of these two types is found, one or more of these patterns - Partner Screening, Execution Monitoring and Penalty and Incentive - can be applied to the control problem in question.

A combination of activities of the *Sub-ideal Counter* type with a *Deny Commitment* activity indicates a control problem of the Proper Contracting pattern. Finally, a combination of *Sub-ideal Counter* type with a *Deny Execution* activity signals a control problem of the Execution Confirmation pattern.

The described matches between the types of sub-ideal counter activities and the relevant patterns are summarized in the Matching Template in Table 11.1.

Table 11.1: Matching Template

Sub-ideal activity type	**Pattern**	**Y/N**
Sub-ideal Counter or/and Emerging Sub-ideal	PARTNER SCREENING EXECUTION MONITORING PENALTY INCENTIVE	
Deny Commitment and Sub-ideal Counter Activity	PROPER CONTRACTING	
Deny Execution and Sub-ideal Counter Activity	EXECUTION CONFIRMATION	

Select between multiple patterns. Patterns are selected on the basis of the type of sub-ideal activity which signals the control problem. As already explained, multiple patterns can be implemented for control problems that have emerged through the activities of the type *Sub-ideal Counter* or/and *Emerging Sub-ideal*. The Execution Monitoring, Partner Screening, Penalty and Incentive patterns, as shown in Table 11.1, can therefore be applied. By using the following rules a further selection can be made:

- Partner Screening involves verification of the counter actor ex-ante. This pattern should be applied for controls before a contract between a primary and counter actors is settled.
- Execution Monitoring suggests the ex-post verification of the counter actor (i.e. monitoring). Tt considers verification of a sub-ideal counter activity and should be applied for controls after a contract between the primary and counter actors is settled.
- The Penalty and Incentive patterns are variations of the Execution Monitoring pattern. They should be applied to add contractual elements of controls, which are missing in the Execution Monitoring pattern.
- The Penalty pattern can be chosen to design a mechanism that punishes a counter actor in order to discourage him from performing sub-ideal behaviour.
- The Incentive pattern can be chosen to design a mechanism that rewards a counter actor and encourages him to behave ideally.

The described choices between the different patterns are shown in Table 11.2.

Table 11.2: Pattern Selection Template

Pattern	**Ex-ante/Ex-post**	**Contractual/Procedural**	**Y/N**
PARTNER SCREENING	ex-ante	procedural	
EXECUTION MONITORING	ex-post	procedural	
PENALTY	ex-post	contractual (punishment)	
INCENTIVE	ex-post	contractual (reward)	

11.3.3 Guidelines

Guideline 4: Treating multiple sub-ideal activities. The primary and counter activities can also be collections of multiple operating activities, as defined in UML [Fowler and Scott, 2000]. In addition, these activities can be executed by an actor, other than the one who transfers the value object. Therefore, there can be more than one sub-ideal activity.

If multiple sub-ideal activities are encountered, any of these activities can be used as an anchor to apply the pattern: several patterns can be applied to control every sub-ideal activity. We advise selecting one sub-ideal activity to proceed and the remaining steps should be repeated for each sub-ideal activity in another e^3*control* cycle.

The selection of a sub-ideal activity is case-dependent and should be made by the user of the methodology since it can only be based on knowledge of the case study. An exemplary choice is to select an activity, verification of which involves the lowest costs. In further steps we consider that **one sub-ideal activity** is selected.

Guideline 5: Delegation of operating activities. The identification of primary and counter activities should take into consideration the fact that activities may have been **delegated** by the primary and counter actors. In such cases, they are executed by an actor other than the one who transfers the corresponding value objects. The primary (counter) activity is not executed by the primary (counter) actor. In the case when primary or counter activities are delegated, Step 3 should be performed in the same way as in the situation without delegation.

Guideline 6: No match. Patterns are selected on the basis of the type of sub-ideal activity which signals the control problem. Four types of sub-ideal activities have been identified: Sub-ideal Counter, Emerging Sub-ideal, Deny Commitment and Deny Execution. If the selected sub-ideal activity does not match any of these four types, two things can be done. Firstly, go back to Step 2 and try redefining the control problem so that it matches one of the sub-ideal activity types. Secondly, the type of control problem may not have been described by any of the patterns in which case the library of control patterns should be extended. An obvious example of such a case is when a control problem is related to information exchanges, e.g. security problems, or a one-off problem. Patterns consider problems at value and process levels, and do not include problems related to information exchanges.

11.3.4 Example

We identify the ideal and sub-ideal primary and counter actors, value objects, and activities for our example.

Primary Actor. In our example, the role of the primary actor is played by the *seller*. The seller does not trust the buyer and expects him to claim wrong or no delivery without good reasons for doing this.

Counter Actor. The counter actor is the *buyer*, and is the one who the primary actor expects to behave sub-ideally.

Sub-ideal Counter Value Object. As can be seen from the sub-ideal value model in Figure 11.3, a sub-ideal counter value object is the object *No Goods Fee*. It is modelled with a dashed line and is exchanged by the buyer, the counter actor.

Ideal Counter Value Object. As can be seen from the ideal value model in Figure 11.2, the *No Goods Fee* replaces the ideal value object *Goods Fee*. *Goods Fee* is therefore the ideal primary value object.

Ideal Primary Value Object. In return for the *Goods Fee* the seller transfers the object *Goods*. This is the primary value object and is illustrated in the ideal value model in Figure 11.2.

Sub-ideal Primary Value Object. In the sub-ideal value model in Figure 11.3, the primary object *Goods* remains ideal. Therefore, there is no sub-ideal primary value object.

Primary Activities. The transfer of the value object *Goods* requires the execution of the activities *Ship Goods* by the seller and *Deliver Goods* by the carrier in the ideal process model in Figure 11.2. Thus, *Ship Goods* and *Deliver Goods* are the primary activities.

Sub-ideal Primary Activities. Since there is no sub-ideal primary value object in our example, we do not have any sub-ideal primary activities.

Counter Activities. The transfer of the value object *Goods Fee* requires the execution of the *Pay* activity by the buyer in the sub-ideal process model in Figure 11.3 and so, *Pay* is the counter activity.

Sub-ideal Counter Activities. Sub-ideal counter activities can be found in the sub-ideal process model in Figure 11.3. The sub-ideal activity *Claim Wrong/No Goods* is an activity of type *Deny Execution*. We also encounter the sub-ideal activity *Pay*, which is not executed and is therefore not shown in the sub-ideal model.

In Figure 11.4 we annotate the elements of the ideal and sub-ideal process model. We indicate which role each element plays in *italics*. In (a) we show the ideal model and in (b) we show the sub-ideal model.

As discussed previously, two sub-ideal counter activities have been found: *Claim Wrong/No Goods* of type *Deny Execution* and non-executed activity *Pay* of type *Sub-ideal Counter*. According to the Matching Template in Table 11.1, these sub-ideal activities indicate the problem of the Execution Confirmation pattern, which we select for further implementation.

11.3.5 References

The execution of Step 3.1. requires knowledge of patterns vocabulary which can be found in Chapter 7. Step 3.2. also requires some acquaintance with the control patterns library as described in Chapter 8 and 9.

11.4 Step 4: Apply the control pattern

The solution to a selected pattern should be implemented by revising the ideal value model and the ideal process model. This involves the following steps.

11.4.1 Step 4.1. Match context and problem

Match the context models of the pattern with the ideal value and process models. In the case of value models, this means matching *actors* and *value objects* of the pattern's context models to the value objects in the ideal process and value models. At the process level it means matching *operational activities* and *objects* of the pattern's context models to operational activities and objects in the ideal process model. The problem value and process models should be matched in a similar way with the sub-ideal value and sub-ideal process models.

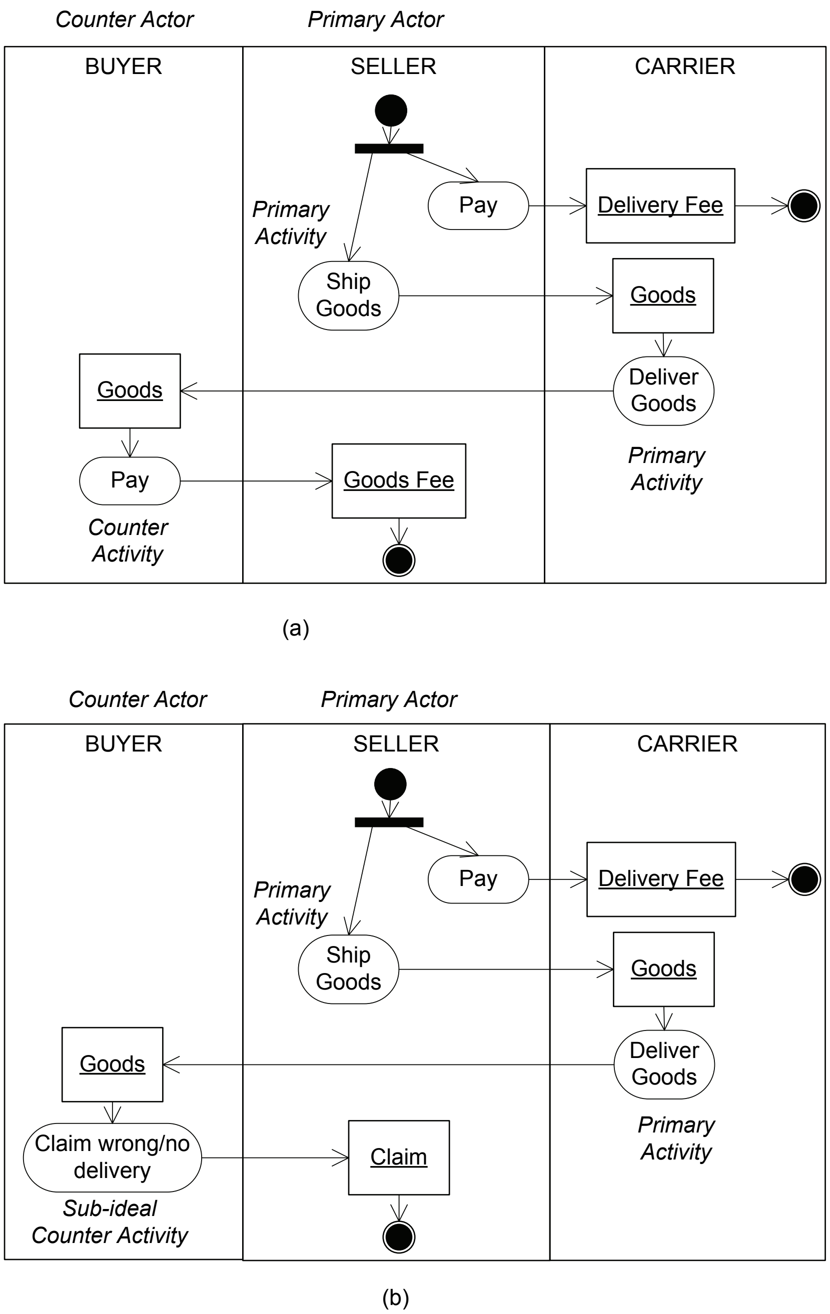


Figure 11.4: An example of Step 3

This match has been partially performed in Step 3, but it only identifies primary/counter activities and primary/counter value objects. The context and problem of the pattern may contain other activities and objects, e.g. an activity *Promise*. All the actors, activities and (value) objects of the context and problem should be identified in this step.

11.4.2 Step 4.2. Design a value model that solves a control problem

The ideal value model should be adjusted by using the suggestions of the value-level *Solution* of the selected pattern:

- Add new actors to the ideal value model;
- Add new value objects to the ideal value model.

The new actors and value objects are suggested by the solution given by the selected pattern. Note that for procedural patterns, described in Chapter 8, the ideal value model remains unchanged unless there is some delegation of control activities.

11.4.3 Step 4.3. Design a process model that solves a control problem

The ideal process model should be adjusted by using the suggestions in the process-level *Solution* of the selected pattern:

- Add new activities and objects to the ideal process model by following the process view of the pattern's solution. In the graphical representations of the patterns in Chapter 8 and in Chapter 9, these new elements are marked with grey.
- Connect the new objects and activities with each other and with other elements of the process model so that it matches the solution given by the pattern.

The new activities and objects are suggested by the pattern's solution. The order of activities and their assignment to the actors are also described by the patterns.

11.4.4 Guidelines

Guideline 7: Starting with a value or a process viewpoint. For procedural patterns, it is convenient to commence application of the pattern at the process level and then change the value model accordingly. For contractual patterns Penalty and Incentive it makes sense to start solution design with a value model. The core of contractual patterns is a change in a value model. The process model might not even be of interest.

Guideline 8: Control Principles. The process model of the solution must comply with the control principles of the applied pattern. If any control principle has not been conformed to, Step 4 must be repeated until all control principles have been observed.

The same goes for the value model of the solution, which must corresponds to the pattern. If this is not the case, then a control weakness is obvious at the value level.

Compliance with the Actors-Activities, Activities-Activities and Activities-Object control principles of the pattern for process models must be checked. For instance, correct segregation of duties must be observed, the execution of witnessing of a counter activity before it occurs must be avoided as should the indirect transfer of documents where it is required.

A Control Principles Checklist can be used to check compliance with the control principles, as in Table 11.3 for the Execution Confirmation pattern. Y should be assigned for each principle in the last column of Table 11.3 if the solution process model complies with the principle. Otherwise, N should be assigned. A control weakness exists if at least one principle is assigned with N. The control principles lists for other patterns can be found in Appendix A.

Table 11.3: Control principles of pattern 'Execution Confirmation'

Principle ID	**Control principles**	**Y/N**
	Activities-Activities principles	
V-Ia	Witness activity must exist	
V-II	Verify activity must exist	
V-IIIa	Witness activity must be executed at the same time as Primary activity	
V-IVa	Verify activity must follow Witness activity	
T - I	Testify activity must exist	
T - II	Testify activity must follow Primary Activity	
PE - I	Verify activity must precede the Testify activity	
	Activities-Objects principles	
V-V	Supporting statement must be directly transferred to Verify activity from a source that generates it	
V-VIa	To-be-verified statement must be directly transferred from Witness activity to Verify activity	
T-III	The outcome of the Verify activity (Positive statement or Negative Statement, both not shown in the model) must be directly transferred from Verify activity to Testify activity	
T-IV	Testifying Statement must be directly transferred from Testify to the primary actor	
	Activities-Actors principles	
V-VIIa	An actor executing Witness activity must be independent and socially detached from the actor executing Counter activity	
V-VIII	An actor executing Verify activity must be independent and socially detached from the actor executing Counter activity	

V-IX	Supporting statement must generated by an actor independent and socially detached from the actor executing Counter activity	
T-V	Testify activity must be executed by the same actor who receives the Positive Statement from the Verify activity	

Guideline 8: Delegation of control activities. The procedural patterns prescribe the execution of control activities to a primary and a counter actor, however, the primary actor or counter actor may not be able to execute them. In such cases, control activities can be assigned to third parties, which requires delegation of the control activities. In such cases, a pattern is applied in a similar way except that the delegated activities are assigned to actors other than the counter or primary actor. In addition, the delegation principles of the applicable delegation patterns must be observed (see Chapter 10).

Guideline 9: Reusing parts of the existing process. In practice, the ideal models often contain some elements of the applied pattern. For example, the ideal process model may contain a *Witness* activity, but not contain a *Verify* activity. In such cases, a pattern is applied in a similar way except that the elements of the pattern are assigned to the existing activities and objects. The emphasis of the application of the pattern lies on verification of the control principles rather than on the introduction of new elements suggested by the patterns.

11.4.5 Example

The context of the pattern is matched to the ideal value and process model, as described in Table 11.4. The left-hand column of the table names elements of the Execution Confirmation pattern. The upper part lists the elements of the context and the lower part lists the elements of the problem. The right-hand column indicates corresponding elements of the ideal and sub-ideal value and process models in Figures 11.2 and 11.3. For example, the first row indicates that the role of the primary actor in the context is played by the Seller in the ideal model in Figure 11.2. The last row shows that the sub-ideal counter activity described in the problem of the pattern is instantiated by the *No Pay* activity in the sub-ideal process model in Figure 11.3.

The Execution Confirmation pattern, selected in Step 3, suggests that the seller must receive a *Testifying Statement* from the buyer. In addition, the pattern suggests that the buyer witnesses and verifies delivery before testifying to it. Finally, *Deny Execution* activity must be executed in the case of negative verification. Therefore, we should add activities *Witness*, *Verify*, *Deny Execution*, and *Testify* and objects *To-Be-Verified Statement*, *Supporting Statement*, and *Testifying Statement* to the ideal process model in Figure 11.2.

The result of the application of the solution to the process and value model is shown in Figure 11.5. The *To-Be-Verified Statement* is represented by a *Delivery Report (DR)* generated after the buyer witnessed the goods. The *Supporting Statement* is a *Purchase Order (PO)*, which states which goods were delivered. If the PO and DR correspond to each other, the buyer pays and testifies to the delivery by executing the activity *Testify*. The testifying statement is represented

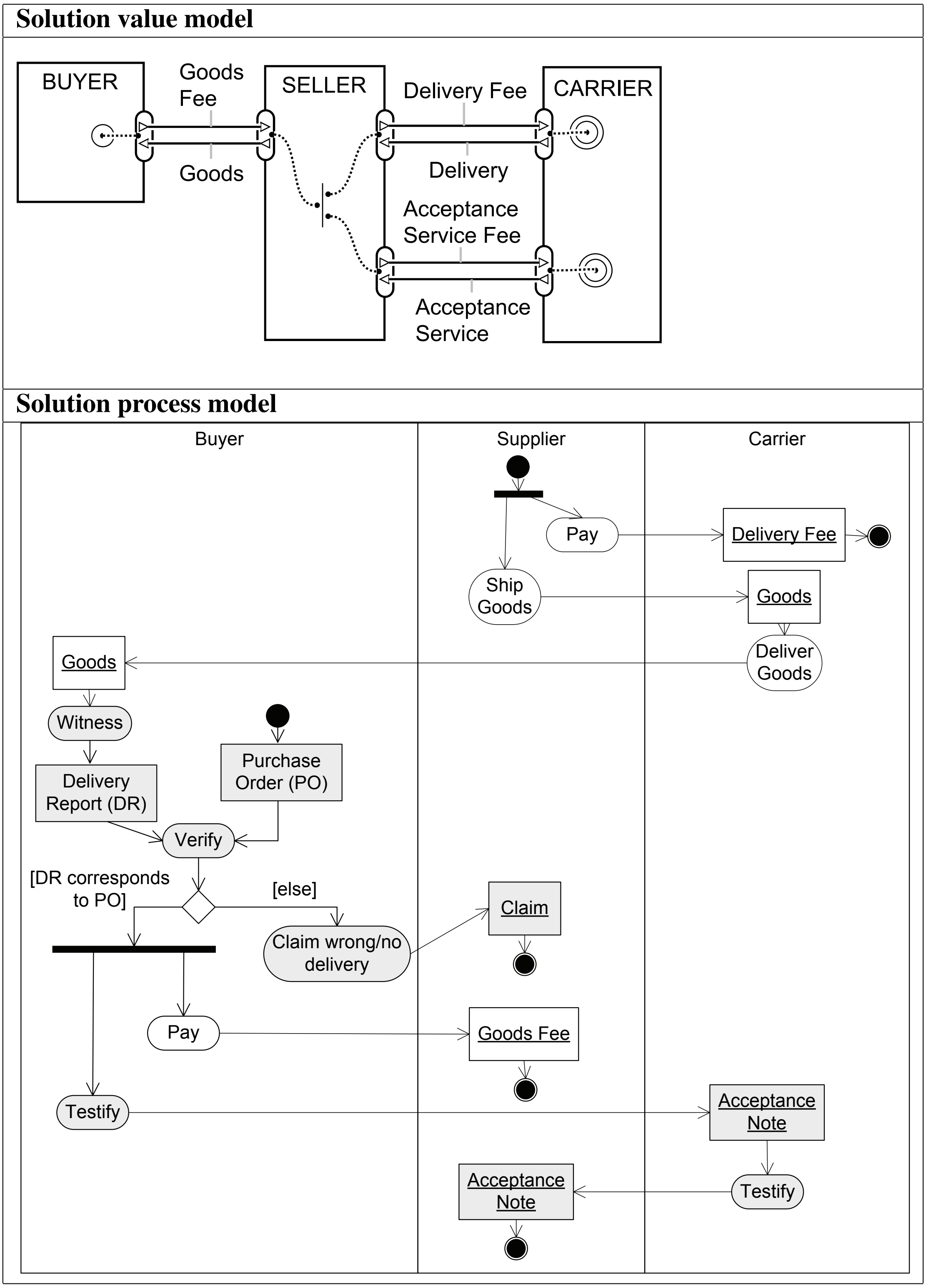


Figure 11.5: An example of step 4

Table 11.4: Matching context and problem

Context	**Ideal models**
Primary Actor	Seller
Counter Actor	Buyer
Primary Value Object	Goods
Counter Value Object	Goods Fee
Primary Activity	Ship Goods (by Seller) Deliver Goods (by Carrier)
Counter Activity	Pay
Problem	**Sub-ideal models**
Primary Actor	Seller
Counter Actor	Buyer
Primary Value Object	Goods
Counter Value Object	Goods Fee
Primary Activity	Ship Goods (by Seller) Deliver Goods (by Carrier)
Deny Execution	Claim Wrong/No Delivery
Sub-ideal Counter Activity	No Pay (by Buyer, not shown in the model)

by *Acceptance Note*, by which the buyer testifies to delivery. After signing the acceptance note, the buyer can no longer deny delivery.

The solution also involves the Testifying Chain pattern, because the *Acceptance Note* is not sent directly to the seller, but to the carrier. This complies with the pattern, in which the carrier is *independent and socially detached* from the buyer. The carrier then testifies to the delivery to the seller by sending the *Acceptance Note* to him.

Note that the resulting model complies with the control principles of the Execution Confirmation pattern. They are listed in Table 11.3.

Since the process-level solution involves delegation of the activity *Testify* by the seller to the carrier, the ideal value model has to be changed correspondingly. In value terms, the carrier delivers a service to the seller by requiring an acceptance note from the buyer and testifying to the delivery to the seller. We call this service *Acceptance Service*. For example, such a service may require the carrier not to give the buyer the goods unless the buyer signs the acceptance note. In return for the acceptance service, the seller pays a separate *Service Fee* to the carrier as shown in the value model in Figure 11.5. The processes that correspond to the payment of the service fee are not yet in the process model in Figure 11.5 and have to be added.

11.4.6 References

Execution of this requires knowledge of the control patterns library, which is described in Chapter 8 and 9.

11.5 Summary

In this chapter we have proposed a structured step-wise approach to the design and redesign of control mechanisms in complex business networks using patterns. The design process starts with designing an ideal situation in step 1 and a sub-ideal situation in Step 2. Further in Step 3 the user selects an appropriate pattern to mitigate the control problem identified in Step 2. In step 4 the user designs the solution to the problem by applying the selected pattern. In Step 4 the user also checks compliance of the process model of the solution with the control principles. If any control principle has not been conformed to, Step 4 must be repeated until no control weaknesses are left. In this section, we have also discussed such issues as multiple control problems, multiple sub-ideal activities, how to match a control problem to a pattern, and how and at what step in the design process to handle delegation issues.

Chapter 12

Renewable electricity in UK

One of the industries with interesting complicated control problems is the renewable electricity industry. In order to comply with international environmental agreements, such as the Kyoto protocol, governments must ensure that a sufficient amount of electricity is produced with technologies that do not use fossil fuel. Examples of CO_2-friendly technologies are wind turbines, photovoltaic panels and hydro generators. Such technologies are called *renewable* or *green* technologies. At present these technologies require high initial investments, which means that the price of green electricity is higher than the price of electricity produced in the conventional way using fuel-based technologies [Laresgoiti et al., 2004]. Many government regulated schemes have been implemented to make renewable technologies commercially more attractive, e.g. tax cuts and subsidies on initial investments, premiums for generated electricity, etc. In this chapter we examine more closely one such scheme which was implemented in the United Kingdom (UK).

In the UK, the Renewable Obligation (RO) regulation law was introduced to stimulate the generation of renewable electricity. The first Renewable Obligation regulation in the UK came into force in April 2002. The law places an obligation on electricity **suppliers**, licensed to supply electricity in the UK, to source a certain proportion of electricity from renewable sources [OFGEM, 2004b]. When the regulation was introduced, this portion constituted 10% of the total supply of a UK supplier. In 2006/07 a UK supplier is obliged to generate 6.7% of its supply from renewable sources.

Suppliers prove that they meet their obligations by presenting Renewable Obligation Certificates (ROCs) each one of which represents one Mega Watt/hour (MWh) of produced renewable electricity output. ROCs can be acquired by suppliers from **producers** of green electricity. The producers get ROCs from a government agency, the Office of Gas and Electricity Markets (**Ofgem**) for each MWh of renewable electricity output they produce. In addition, Ofgem maintains a register of all ROCs it has issued.

The suppliers must therefore provide ROCs as evidence of how much MWh of green electricity they have supplied. If a supplier does not have sufficient ROCs to cover his obligation, he must make a deposit into a *buy-out fund*. The buy-out fee is a fixed price per MWh shortfall and is adjusted in line with the Retail Prices Index each year. Premiums from the buy-out fund are paid back to suppliers in proportion to how many ROCs they have presented.

In this case study we apply the e^3*control* methodology and patterns to describe the controls that appear in this case. Actually, we reverse engineer the ultimate ROC-scheme with the goal of explaining, by means of the patterns, why this scheme is needed from the control perspective. For example, we illustrate how the patterns explain the necessity of introducing ROCs.

Our discussion starts in a situation of no ROCs introduced. We then go on to explain the control problems that may occur in such a network. We explicitly take the government's point of view and only describe the problems as perceived by the government, which is represented by Ofgem. Then, step by step, we design the ROC scheme by applying the patterns. As a result, we will demonstrate that the ROC scheme can be explained by means of e^3*control* patterns.

12.1 Research Context

The initial data collection for the case study was carried out within the EU-funded research EC-EESD project BusMod[1] before this research started. During the BusMod project, the data was collected by several researchers, including the author of this thesis. It mostly took the form of organized brainstorming sessions with domain experts from different organizations in the network. Domain experts from six different European organizations participated in the BusMod project. The advantage of such a setting is that it was possible for us to actively participate in the project meetings with the domain experts. One of the primary goals of these meetings was to share knowledge on how the electricity markets operate in different countries, including the UK. During the BusMod project we also had a chance to verify some results of the value modelling with the domain experts. For example, we gave a workshop during which the domain experts learned how to build e^3*value* models. In fact, the models presented here are based on another value model of the UK electricity market built by the domain experts themselves. This latest model can be found in [Mutale and Strbac, 2004].

Knowledge about processes behind the ROC scheme was gained mostly from material found on the Ofgem web site (www.ofgem.co.uk), including [OFGEM, 2004a], [OFGEM, 2004b] and [OFGEM, 2006].

12.2 The first e^3*control* cycle: Non-tradable ROCs

12.2.1 Step 1: Ideal situation

Value view. Figure 12.1 presents an ideal value model for the renewable energy case. In this model we show desirable behaviour of the suppliers from the perspective of Ofgem. As was explained earlier, Ofgem wants the suppliers to provide at least 10% of their electricity from renewable sources.

The dependency path in Figure 12.1 starts at the **customer**, the final electricity consumers in the UK. The customer buys *Electricity* from the **supplier** and pays the supplier a *Retail Fee* in

[1] http://www.e3value.com/projects/ourprojects/busmod/

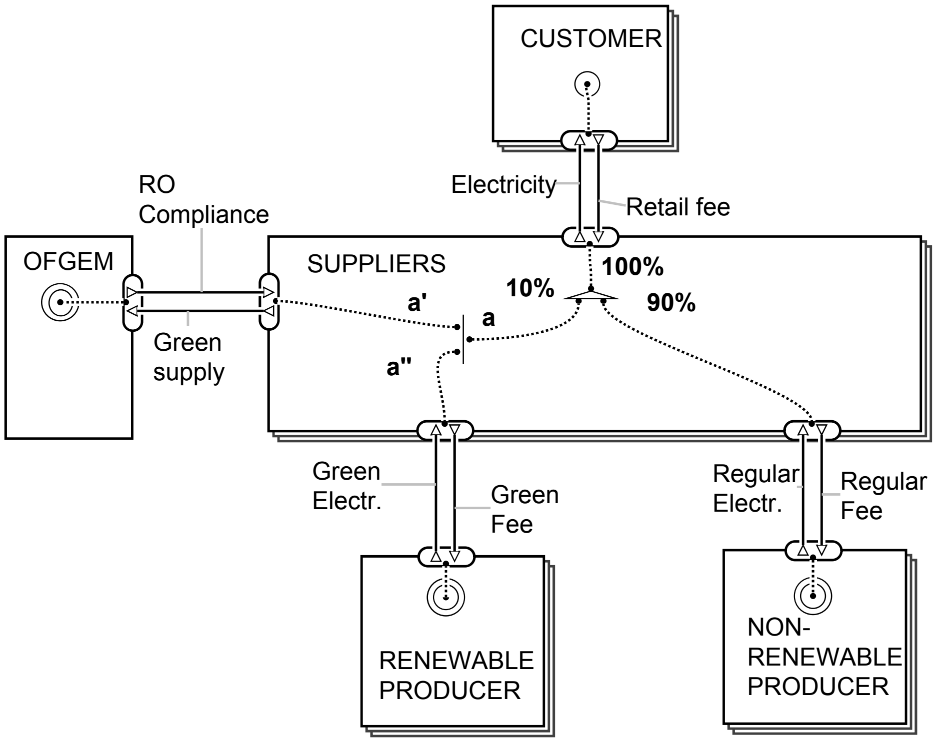


Figure 12.1: An ideal value model of the ROC business model

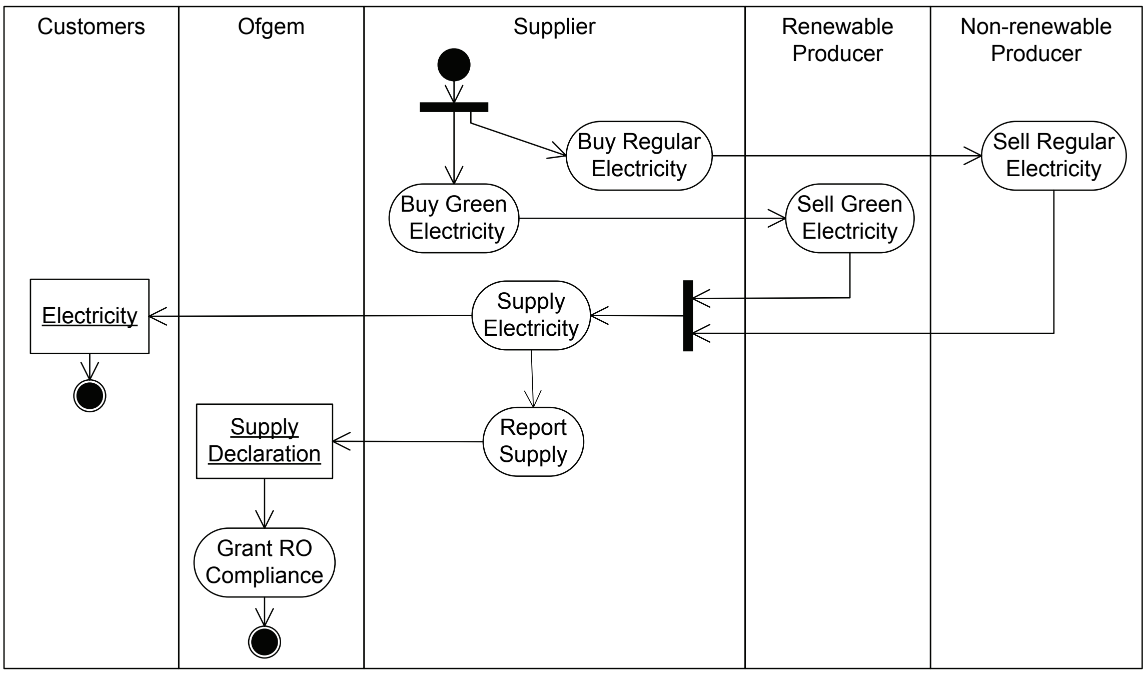


Figure 12.2: The ideal process model of the ROC business case

return[2]. As denoted by the OR-fork at the supplier, the supplier can buy electricity from two sources: from **non-renewable producers** or from **renewable producers**. For more specific definition of what producer falls under renewable producers see regulation in [OFGEM, 2004b]. In the first case, the supplier buys *Regular Electricity* and pays *Regular Fee* in returns . In the second case, the supplier buys *Green Electricity* and pays a *Green Fee*. Because green electricity is produced by more expensive renewable technology, the renewable producer asks a higher price for electricity than the non-renewable producer. Therefore, we assume that the following inequality holds:

$$RegularFee < GreenFee$$

According to the RO regulation, a supplier has to obtain 10% of electricity from renewable sources[3]. In e^3*value* terms, this means that the electricity delivered by buying *Green Electricity* in Figure 12.1 has to account for at least 10% of the *Electricity* supplied to the customers.

In this model, we assume that the suppliers behave ideally and always buy 10% of their supply from renewable producers. Therefore, when a supplier buys *Green Electricity*, he simultaneously reports the supply of green electricity to **Ofgem** and receives a compliance with the renewable obligation. This is modelled with objects *Green Supply* and *RO Compliance* accordingly (see path *a'*).

Note that the value objects *Green Electricity* and *Green Supply* are two different value objects. *Green Electricity* denotes an amount of electricity, whereas *Green Supply* denotes the fact of generation of electricity with *renewable* sources.

Process view. In Figure 12.2 we represent an ideal process model that corresponds to the described ideal value model. The process starts at the supplier who, as in the value model, has the choice of buying electricity from a renewable or a non-renewable supplier. In the first case, the supplier executes a *Buy Regular Electricity* activity, followed by a *Sell Regular Electricity* activity of the non-renewable supplier. In the second case, the suppler executes a *Buy Green Electricity* activity, followed by a *Sell Green Electricity* activity executed by the renewable supplier. After that, in both cases a *Supply Electricity* activity is executed by the supplier, which results in a transfer of an object *Electricity* from the supplier to the customer. Further, the supplier reports information about his supply (in MWh) to Ofgem by transferring a statement *Supply Declaration* to Ofgem. In the *Supply Declaration*, the supplier reports how much green electricity was supplied and what part of this electricity was green.

Since this model represents an ideal situation, the supplier is always assumed to behave ideally. In other words, the suppler always buys at least 10% of green electricity. Therefore, at the end of the process the *RO compliance* is always granted.

[2]In this model, the customer buys both green and conventional electricity for the same price. However, other business models are possible, in which the customer is charged a higher fee for green electricity than for conventional supply.

[3]When the regulation was introduced in 2002, the limit was around 10%. Currently in 2006/07 it is 6.7% and 2.6% in Northern Ireland. In this model we assume a limit of 10%.

12.2.2 Step 2: Sub-ideal situation

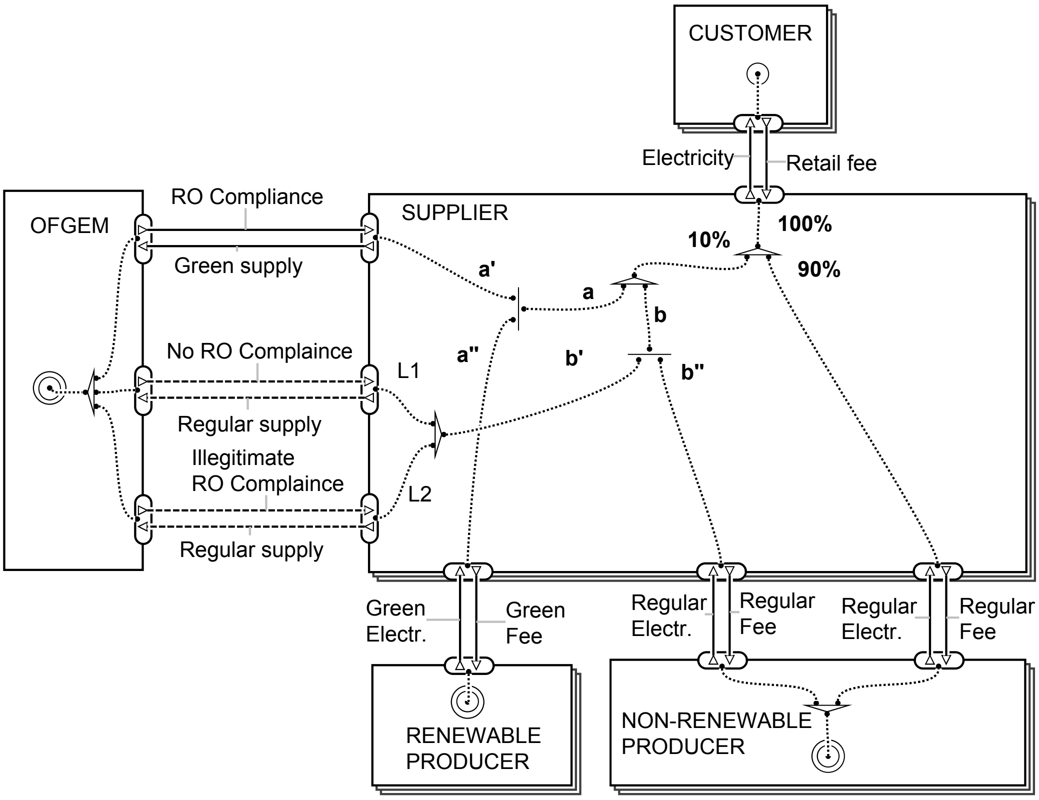


Figure 12.3: Sub-ideal value model for the problem of not supplying green electricity

We can identify *two types of sub-ideal behaviour* of suppliers within the developed models. Firstly, not every supplier complies with the renewable obligation, and those that do not comply with the RO are considered by Ofgem as behaving sub-ideally. A sub-ideally behaving supplier buys a lower percentage of green electricity than the 10% prescribed by the regulation. In this case, the *RO compliance* is not (completely) granted.

Secondly, some suppliers can *overstate* the percentage of green supply in order to obtain the RO compliance illegally. To do this, the suppliers report a higher percentage of green electricity and get the *Illegitimate RO compliance*. Another way to mislead the Ofgem is to understate the total supply so that the percentage of the green supply constitutes 10%. This also leads to *Illegitimate RO compliance*.

Below we model these two sub-ideal situations. Firstly, we represent the problem in a value model, and then we model the problem in a process model.

Value view. A sub-ideal value model in Figure 12.3 models both the ideal and sub-ideal behaviour of a supplier. The second OR-fork leads to the ideal paths *a* and sub-ideal path *b*. The ideal path *a* shows the same as in the ideal value model: the supplier buys green electricity and gets RO compliance.

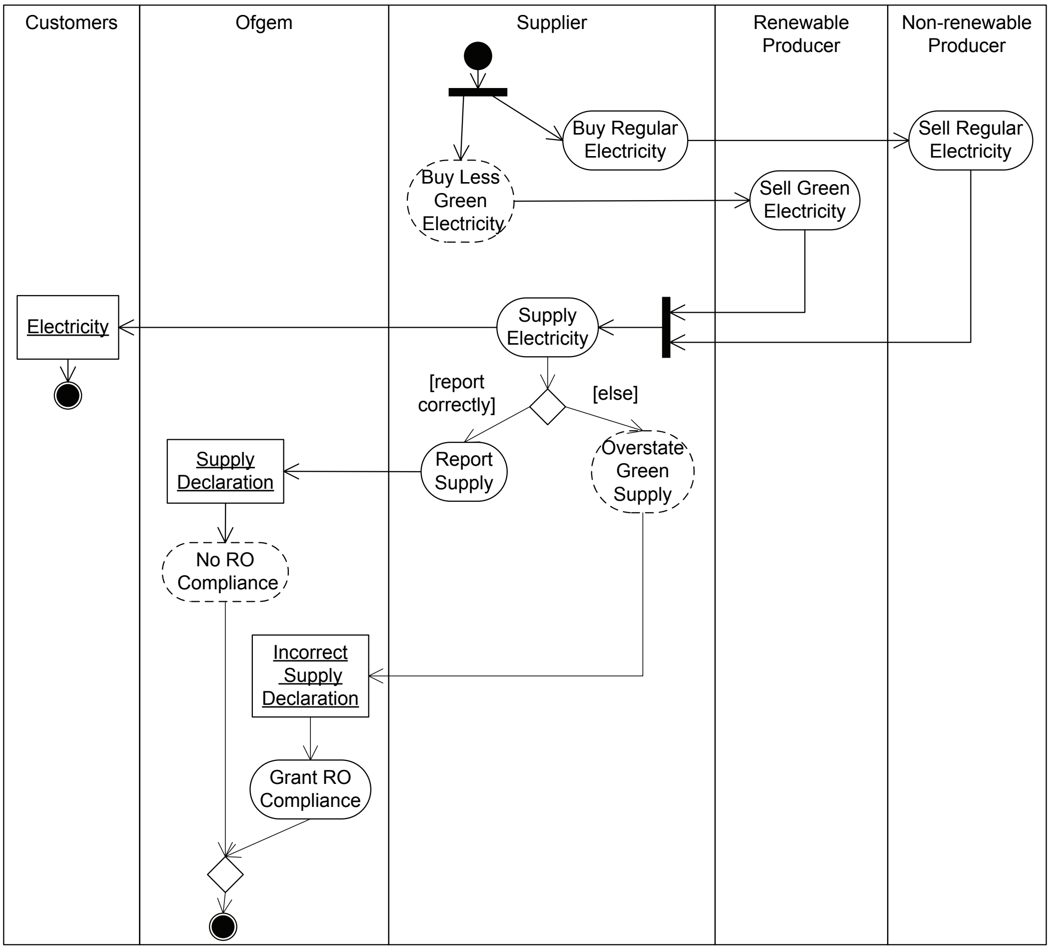


Figure 12.4: Sub-ideal process model for the problem of not supplying green electricity

The sub-ideal path *b* corresponds to the two types of sub-ideal behaviour. In both cases, the supplier buys *Regular Electricity*, instead of *Green Electricity*. This corresponds to the exchanges in the sub-path *b”*. Further, the *OR-fork* at the sub-path *b’* indicates two possibilities of sub-ideal behaviour. The sub-path, marked with a liability token **L1**, corresponds to a situation in which the supplier reports his low supply of green electricity and does not get the RO compliance. At the sub-path, marked with a liability token **L2**, the supplier either overstates his low supply of green electricity or understates the total supply. As a result, he gets the RO compliance illegitimately. The value objects that correspond to this situation are marked as sub-ideal with dashed value transfers.

Process view. In Figure 12.4 we represent only the sub-ideal behaviour of the supplier. As in the ideal process model, the initial state of the sub-ideal process model is located at the supplier who has a choice of buying electricity from the renewable or non-renewable producer. In the represented sub-ideal behaviour, the supplier buys insufficient green electricity. We model it with a sub-ideal activity *Buy Less Green Electricity* instead of the *Buy Green Electricity*, as in the ideal process model.

After supplying the electricity the supplier has a choice of reporting the true supply of green electricity or of overstating it. The choice is modelled with the UML decision element. In the first case, the supplier transfers a *Supply Declaration* in which he informs Ofgem about insufficient green supply and, as a result, he does not get the RO compliance. In the second case, the supplier overstates the percentage of green supply and transfers an *Incorrect Supply Declaration*. As a result, the supplier gets the RO compliance illegitimately.

12.2.3 Step 3: Select a control pattern

In order to solve the control problem, Ofgem should implement one or more control mechanisms. In this step we select a pattern that is appropriate for the control problem described in step 2.

Step 3.1. Identify ideal and sub-ideal primary and counter actors, value objects, and activities

To find a pattern, we must first identify the primary and counter actors, as well as the primary and counter activities and value objects. Then, we identify which pattern can be applied to mitigate the selected sub-ideal counter activity.

Primary Actor. The primary actor in this case is *Ofgem*. This is the actor who considers an insufficient green supply to be a problem.

Counter Actor. A counter actor is *Supplier*. He is considered to behave sub-ideally by not supplying sufficient green electricity. In the sub-ideal value model in Figure 12.4 the supplier has liability tokens L1 and L2.

Sub-ideal Counter Value Object. The result of the sub-ideal behaviour of the supplier is the sub-ideal value object *Regular Supply* in the sub-ideal path **L1** and *Overstate Green Supply* in the sub-ideal path **L2**. These are sub-ideal counter value objects.

Sub-ideal Primary Value Object. The value object *No RO Compliance* and *Illegitimate RO Compliance* are the sub-ideal primary value objects. They are transferred in return for the sub-ideal counter value objects identified earlier.

Ideal Counter Value Object. The value object *Green Supply*, which in the ideal value model corresponds to the sub-ideal counter value object, is the ideal counter value object.

Ideal Primary Value Object. The value object *RO Compliance*, transferred in return for the ideal counter value object *Green Supply* is the ideal primary value object.

Primary Activities. To transfer the primary value object *RO Compliance* the Ofgem executes the activity *Grant RO Compliance*. So, *Grant RO Compliance* is the primary activity.

Sub-ideal Primary Activities. The sub-ideal primary activities are *No RO Compliance* for the sub-ideal path **L1** and *Illegitimate RO Compliance* for the sub-ideal path **L2**.

Counter Activities. The Supplier must execute several activities to transfer the counter value object *Green Supply*. Firstly, he buys green electricity (see activity *Buy Green Electricity*). Secondly, he reports the supply to Ofgem (see activity *Report Supply*). The activities *Buy Green Electricity* and *Report Supply* are therefore ideal counter activities.

Sub-ideal Counter Activities. For the sub-ideal path **L1**, the sub-ideality of the value object *Green Supply* is caused by delivering less green electricity than required. So, the sub-ideal counter activity is *Buy Less Green Electricity*. The sub-ideality of the value object *Green Supply* in the sub-ideal path **L2** is not only caused by delivering insufficient green electricity, but also by overstating the green supply. So, the activity *Incorrect Supply Declaration* is also sub-ideal.

As a result, we have selected two sub-ideal activities: *Buy Less Green Electricity* and *Incorrect Supply Declaration*. To proceed, we must focus on one sub-ideal counter activity and deal with the rest of the activities later on. From now on we focus on the activity *Buy Less Green Electricity*. Later on we show that the pattern that is being used to prevent sub-ideality of the activity *Buy Less Green Electricity* also helps to avoid the activity *Incorrect Supply Declaration*.

Step 3.3: Identify type of sub-ideal activity

To select a pattern, we must first identify the *type* of *Buy Less Green Electricity*, which is the sub-ideal activity under consideration. In Chapter 7, we distinguish three types of sub-ideal activities: Sub-ideal Counter, Emerging Sub-ideal, Deny Commitment and Deny Execution (see Chapter 11 for more details). The sub-ideal activity *Buy Less Green Electricity* is of the first type Sub-ideal Counter. In fact, this sub-ideal activity is a sub-ideal variant of the ideal counter activity *Buy Green Electricity*, but executed in an incorrect way. We do not see activities of types Deny Execution and Deny Commitment in the sub-ideal process model.

Step 3.4: Match the sub-ideal activity to a pattern

Table 12.1: Pattern Selection Template for ROC case

Pattern	**Ex-ante/Ex-post**	**Contractual/Procedural**	**Y/N**
PARTNER SCREENING	ex-ante	procedural	N
EXECUTION MONITORING	ex-post	procedural	N
PENALTY	ex-post	contractual (punishment)	Y
INCENTIVE	ex-post	contractual (reward)	Y

The type **Sub-ideal Counter** signals the presence of the control problems described in the Partner Screening, Execution Monitoring, Penalty, and Incentive patterns. Further we use the following steps to select the pattern:

- The Partner Screening pattern considers an ex-ante problem of hidden characteristics, e.g. that a supplier would hide from Ofgem his ability to deliver green energy. This is not the control problem we want to solve, so we do not select the Partner Screening pattern. We limit our choice to the Penalty, Incentive and Execution Monitoring patterns, which consider ex-post problems of hidden action.
- Execution Monitoring suggests the ex-post verification of a sub-ideal counter activity *Buy Less Green Electricity*. The solution given by the Execution Monitoring pattern suggests identifying parties that supply less green energy and not grant them RO compliance. However, this will not harm the suppliers much, unless a refusal to grant RO compliance is supported by a monetary loss for the suppliers. This monetary loss can be introduced by Penalty and Incentive patterns, which we select as a solution to our problems.
- The Penalty and Incentive patterns are variations of the Execution Monitoring pattern. The Penalty pattern can be used to punish suppliers for delivering less green electricity than required. The Incentive pattern can be used to reward them for buying green electricity.

In the ROC scheme, both Penalty and Incentive mechanisms are implemented. The final choice is summarized in the Pattern Selection Template in Table 12.1.

12.2.4 Step 4: Apply the Penalty and Incentive patterns

In order to design a solution, we need to add elements of the solutions given by the selected Penalty and Incentive patterns to the ideal value and process models shown in Figures 12.1 and 12.2. As a result, we obtain the solution value and process models shown in Figures 12.6 and 12.5 correspondingly.

Process view. The process level solution of both Penalty and Incentive patterns requires the primary actor Ofgem to check the outcome of the sub-ideal counter activity *Buy Green Electricity* and, depending on the outcome, to reward or punish the supplier. To model this, we first add a *Verify Compliance* activity to Figure 12.5. This is an instance of a *Verify* activity in the patterns (see Figures 9.1 and 9.3).

Furthermore, according to the pattern, *Verify Compliance* requires two inputs: *Supporting Statement* and *To-be-verified Statement*. The *To-be-verified Statement* gives information about a percentage of electricity supplied from renewable sources. In fact, this corresponds to the object *Supply Declaration*, presented in the model.

According to the pattern, the to-be-verified statement *Supply Declaration* must be produced by an activity that *witnesses* the activity *Buy Green Electricity*. In addition, the pattern requires this witnessing activity to be executed either by the primary actor Ofgem or by a party independent and socially detached from the counter actor Supplier. If this is not the case, the *violation of segregation of duties* occurs, since the supplier who is responsible for buying green electricity also executes reporting it.

In the ideal value mode, the to-be-verified statement *Supply Declaration* is produced by the Supplier. This violates the requirements of the pattern. To correct the situation, the following changes are made in the solution model. Firstly, we explicitly model the witnessing activity *Witness Green Supply*, instead of only the reporting activity. Secondly, we assign this activity to an actor, who is independent from the Supplier and is able to produce trustworthy information about the supply. Ideally such an actor is the primary actor Ofgem. However, as it is just an administrative body, Ofgem does not have the resources to control each supplier. So, Ofgem delegates control of the suppliers to some trusted party.

The role of such a trusted party can be played by the renewable producer who supplies electricity to suppliers. This party is therefore physically able to keep track of how much green electricity is bought by each supplier. In Figure 12.5, the activity *Witness Green Supply* is assigned to the renewable producer.

The third change me make is to rename the to-be-verified statement. This statement, previously called *Supply Declaration*, in now called *Renewable Obligation Certificate (ROC)* in this model. One ROC is issued for each Mega Watt/hour (MWh) of eligible renewable output. According to the pattern, the ROC is fed into the activity *Verify Compliance*, which compares whether the ROCs of one particular supplier represent 10% of his total supply.

The ROC only represents the amount of green electricity. However, the important criteria for granting RO compliance is the *share* of the green electricity within the supplier's total electricity supply. Therefore, we add another to-be-verified statement *Total Supply*, which represents this information. As with *ROC*, the *Total Supply* must be generated by an actor who is independent and not acting in the interests of the supplier. For instance, the data about the total supply could be retrieved from the final customer or from the supplier's annual accounts, assuming they are trustworthy. In the model we show that the *Total Supply* is generated by Ofgem. Data concerning the total supply of each supplier is easily accessible to a governmental organization like Ofgem.

The *Verify Compliance* activity requires a supporting statement, namely, information, which is needed to decide whether the *RO Compliance* should be granted. Such a document is an *RO* *legislation*, stating e.g. the required percentage of green electricity (which we assume is 10%), which producers are qualified to hold the status of 'renewable', to which customers should the reported green electricity be supplied, etc.

In addition, according to the patterns, the activities are assigned in a proper order, as required by the control principles. The activity *Witness Green Supply* is executed after the *Buy Green Electricity* activity and before *Verify Compliance* activity. In addition, the *Verify Compliance* activity is executed before the primary activity *Grant RO Compliance*.

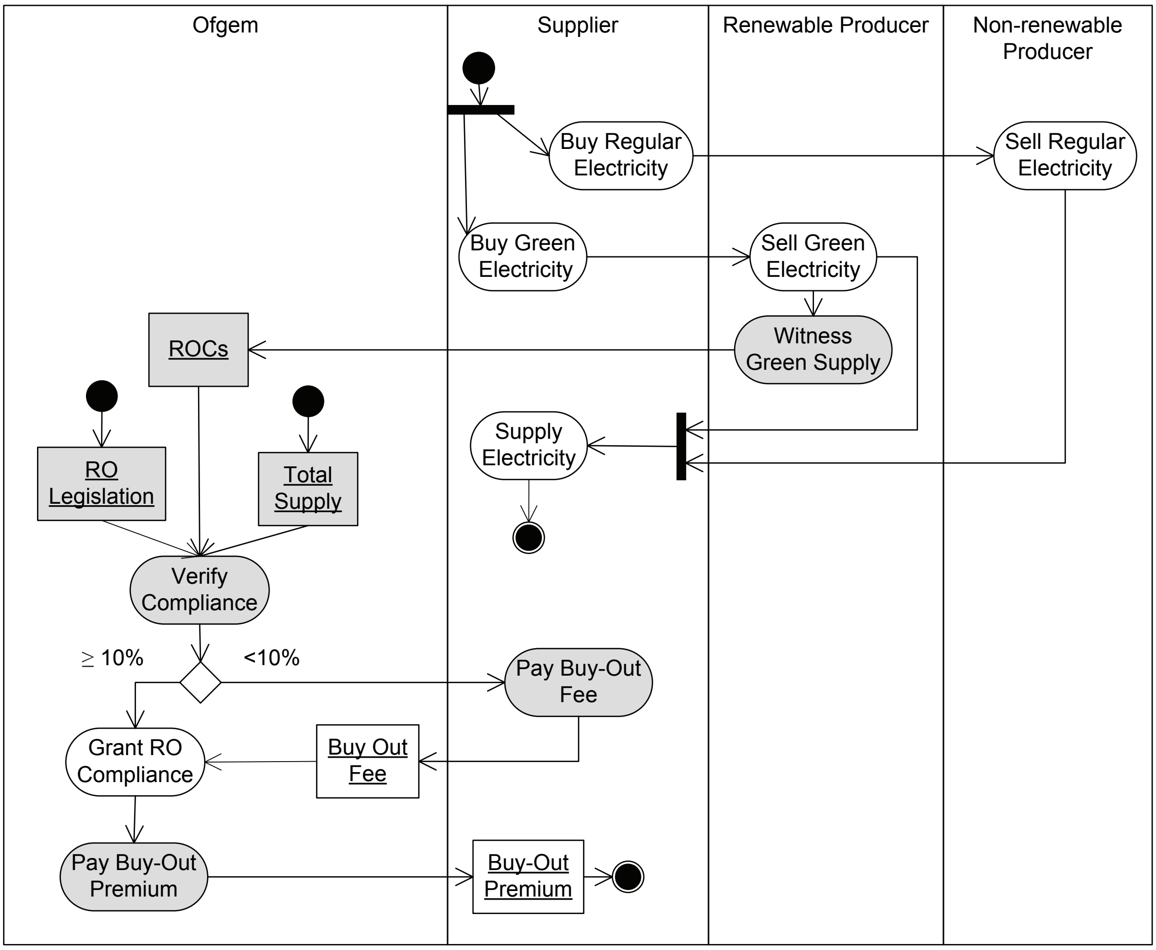


Figure 12.5: Solution process model with penalties and incentives

According to the penalty pattern, an object *Penalty* should be added. The penalty should be transferred to Ofgem by a supplier who supplies less than 10% of green supply. According to the RO regulation, a supplier who does not have sufficient green supply to cover his obligations must make a deposit into the **buy-out fund** [OFGEM, 2004b]. Such a payment corresponds to the object *Penalty* of the Penalty pattern. In Figure 12.5 the *Buy-out Fee* is paid by the suppliers according to the RO regulation. According t the pattern, if the outcome of the *Verify Compliance* states that the green supply is less than 10%, the RO compliance is granted only after the buy-out fee is paid by the supplier. The *Pay BuyOut Fee* activity corresponds to the *Pay Penalty* activity of the Penalty pattern.

According to the solution given by the Incentive pattern, an object *Incentive* should be added. This object should be transferred to the supplier who supplies at least 10% of green supply. In reality, this kind of penalty exists. The buy-out fund is paid back to suppliers in proportion to how much green electricity they have purchased [OFGEM, 2004b]. This payment, henceforth called *Buy-out Premium*, represents the incentive. We add it to Figure 12.5.

In addition, according to the pattern, we model that if the outcome of the *Verify Compliance* states that the green supply is more than 10%, then RO compliance is granted and the *Buy-out Premium* is paid. The *Pay BuyOut Premium* activity corresponds to *Pay Incentive* activity of the Incentive pattern.

Note that in the application of this pattern we have also used the Simple Delegation pattern, see Chapter 10. This is needed to model that the *Witness Green Supply* activity is delegated by Ofgem to the trusted actor Renewable Producer.

Value view. We now make appropriate changes in the ideal value model. The changes have been caused by the introduction of penalties and incentives as well as by the delegation of the witnessing activity.

Penalty. According to the value level solution given by the Penalty pattern, we make the following changes in the ideal value model. We add a new value transfer, indicating a penalty, as an outgoing value object of the counter actor Supplier in the sub-ideal path **L1**. We add it to the transfer of *No RO Compliance* and *Regular Supply*, and change *No RO Compliance* to *RO Compliance*.

In addition, because we have renamed the *Supply Declaration* to *ROC* in the process model, the ROC also appears as a value object in the value model. We model *ROC* instead of the value object *Green Supply* and *No ROC* instead of the value object *No Green Supply*.

The resulting value model is presented in Figure 12.6 and this corresponds to reality. The penalty is represented by a value transfer *Buy-out Fee*. So, at the sub-ideal path *b*, where the supplier does not supply enough green electricity, he is obliged to pay a buy-out fee in order to cover the RO.

Incentive. According to the solution given by the Incentive pattern, an incoming value object *Incentive* should be added to the Supplier in the ideal value transfer. As explained earlier, the incentive is the buy-out premiums paid by Ofgem to compliant suppliers. As a result, the incentive value object *BuyOut Premium* is added to the transfer of *Green Supply* and *RO Compliance* in Figure 12.6.

Witnessing service. The changes at the process level require some additional clarification of the value model. Because the renewable producer executes a witnessing activity on behalf of Ofgem, a new value object, e.g. *Witnessing Service*, should modelled to represent this service. However, in reality, Ofgem most probably does not pay the producer for the witnessing service. Empowered by the status of a government agency, Ofgem obliges the producer to submit this

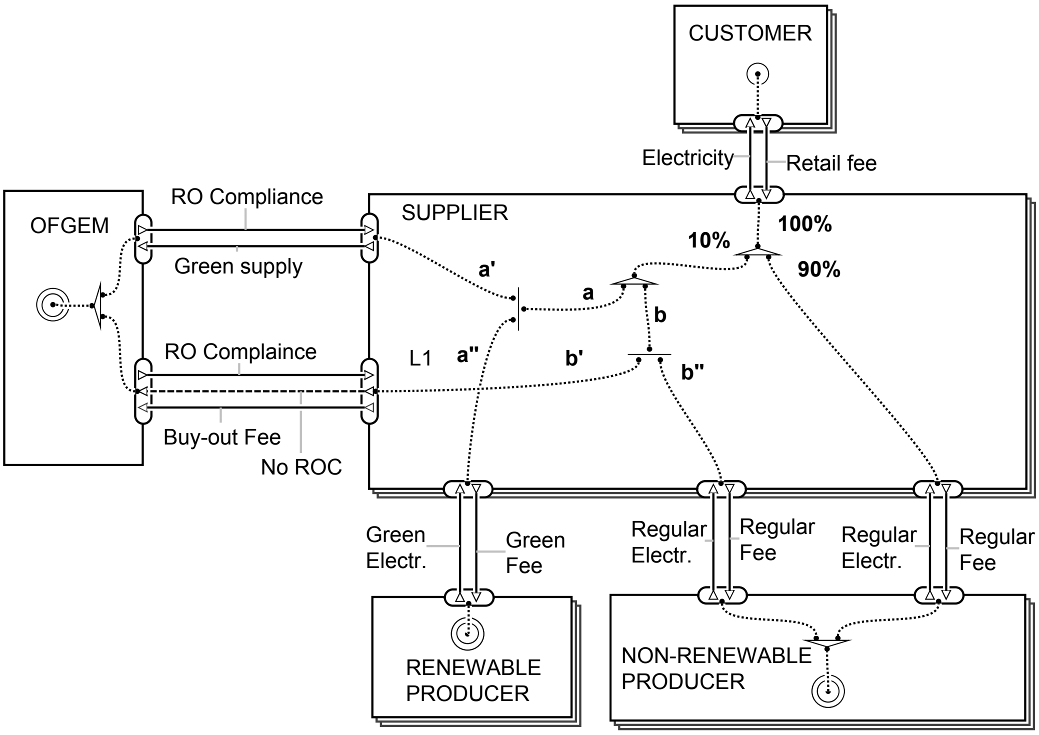


Figure 12.6: Solution value model with penalties and incentives

information. We will not focus on this any further and therefore, we will not model this service in the value model.

12.3 The second e^3*control* cycle: Tradable ROCs

The solution process model in Figure 12.5 represents only a part of the actual process of ROCs scheme. The scenario of transferring ROCs from producers to Ofgem is not completely realistic. Due to the nature of the electricity business, suppliers can buy and sell electricity several times to other suppliers before it reaches the final customer. According to RO regulation, ROCs can be claimed by the supplier who delivers the associated green electricity to final customers. If a supplier sells green electricity to another supplier, the ROCs should also be transferred to this other supplier.

In addition, in practice not only is transfer the ROCs with the associated electricity allowed, but trading them separately from the associated electricity is also allowed. This creates a *market of ROCs*. The ROC is in fact a security similar to stocks and bonds. As will be explained later, the ROC market was created to stimulate green electricity production.

Because ROCs can also be traded amongst suppliers, additional controls are required. We model these controls below in a new e^3*control* cycle. We present this cycle in a short format, paying less attention to all the details of pattern selection and application.

12.3.1 Step 1: Ideal value and process models

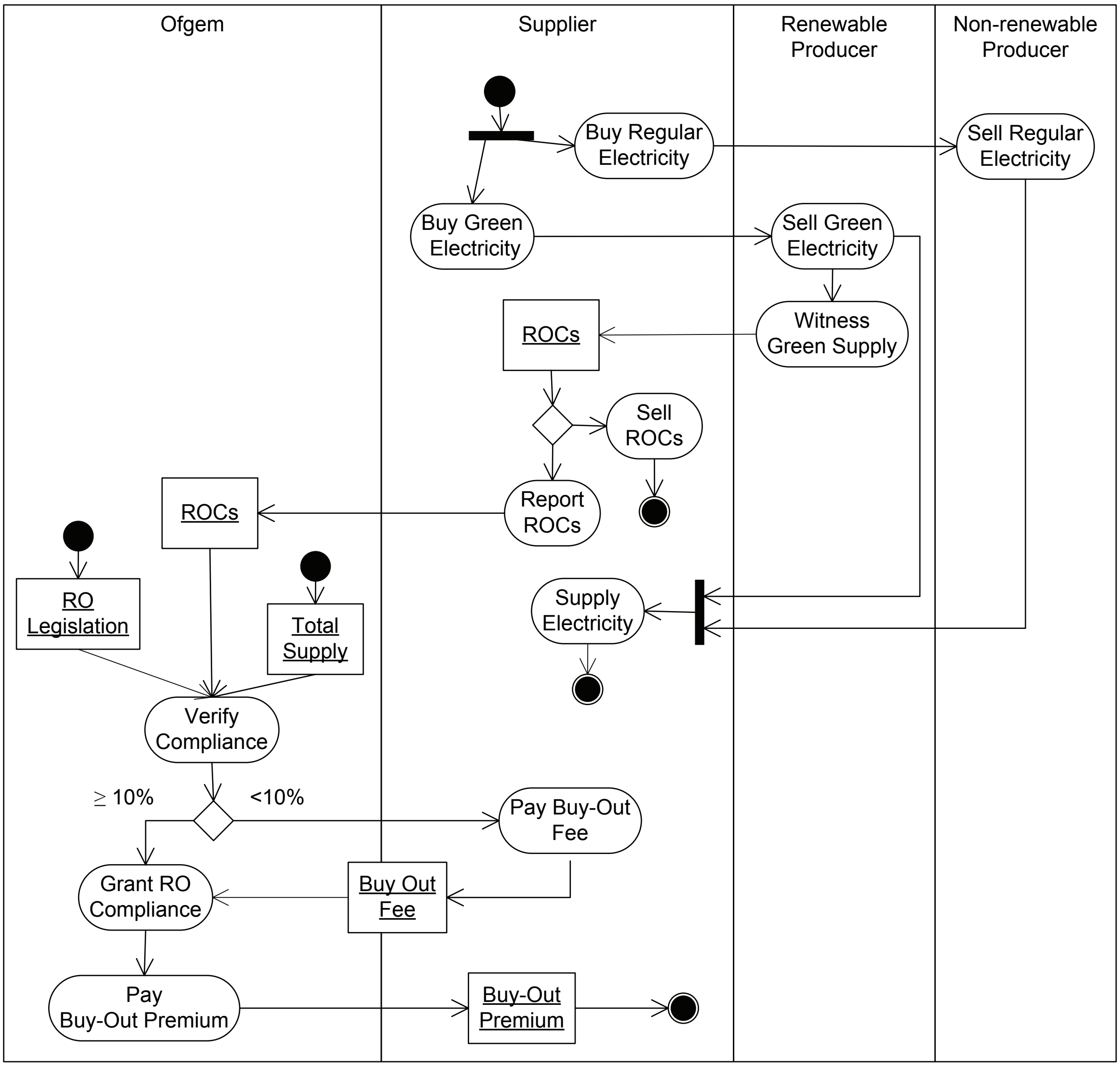


Figure 12.7: An ideal process model with tradable ROCs

The ideal process model of the scenario with tradable ROCs is shown in Figure 12.7. Unlike in the solution process model in Figure 12.5, here the ROCs are transferred to the supplier before being transferred to Ofgem to comply with the required percentage of supplied renewable energy. The supplier has the choice of selling the obtained ROCs or of reporting them to Ofgem. This choice is denoted by the UML decision element at the supplier. If the supplier reports the ROCs to Ofgem, the verification process of charging the buy-out fee or the paying buy-out premium remains the same as before. However, if the supplier sells the ROCs, they are not presented to Ofgem.

The corresponding value model is shown in Figure 12.8. The following changes have been made compared to the situation without tradable ROCs in Figure 12.6. Firstly, suppliers obtain ROCs from the renewable producers. Secondly, unlike in the solution value model in Figure 12.6, the

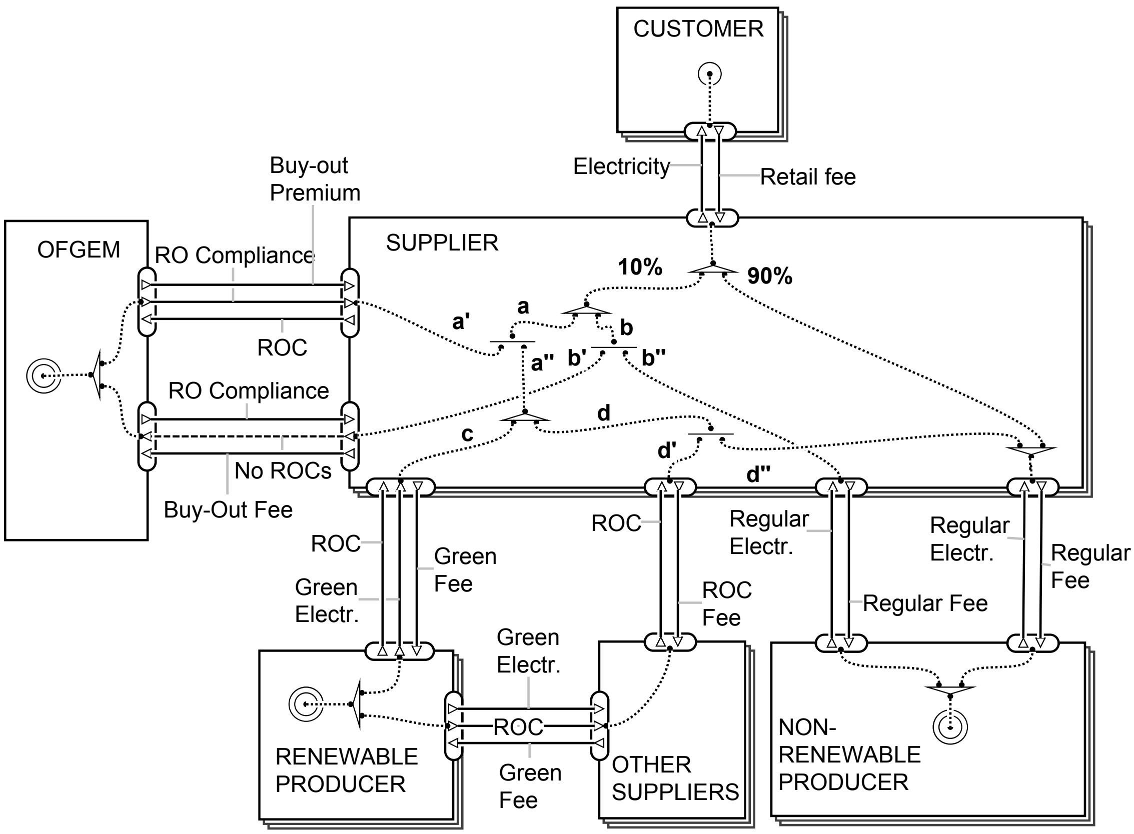


Figure 12.8: An ideal value model with tradable ROCs

ideal dependency path now offers a choice between (1) obtaining ROCs for free while buying the (more expensive) green electricity in path *c* or (2) buying ROCs separately and purchasing the (cheaper) regular electricity in the path *d*. Thus, because ROCs can be traded, they are modelled as value objects, and not as process objects only.

12.3.2 Step 2: Sub-ideal value and process models

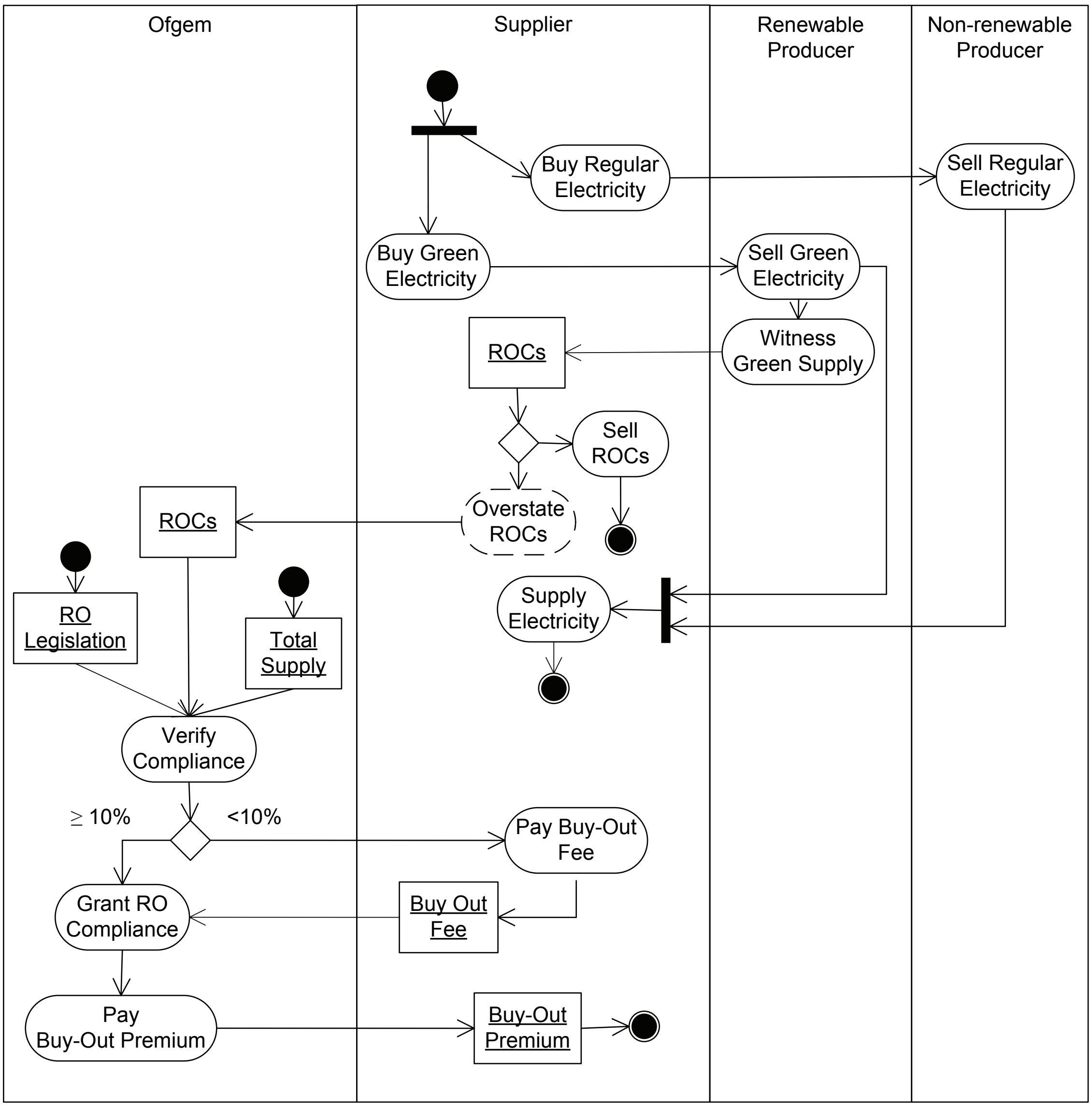


Figure 12.9: A sub-ideal process model with tradable ROCs

The new ideal models in Figures 12.7 and 12.8 differ from the pattern-compliant models in Figure 12.6 and 12.5. In fact, the new models do not comply with the prescriptions of the Penalty and Incentive patterns, applied earlier. To be specific, to perform the *Verify Compliance* activity, Ofgem has to rely on information received from the supplier. This implies that the supplier performs *Witness Green Supply* activity. As already explained, this contradicts the pattern, since the supplier should not report his own activities. For example, the supplier can forge ROCs and overstate the number of supplied green electricity.

In Figure 12.9 we show the sub-ideal process model of the scenario with tradable ROCs. The supplier overstates the number of ROCs he has, which is modelled by an activity *Overstate Green Supply*. This corresponds to the transfer of *No ROCs* by the supplier in the sub-ideal value model in Figure 12.10. Because the overstatement remains undetected, the supplier receives the *RO Compliance* and even gets the *BuyOut Premium* in return. This path is marked with the liability token L3.

Note that the supplier has an illegitimate interest in overstating the number of ROCs which does not depend on whether he can cover the RO obligation or not. If the supplier has enough ROCs to cover the obligation, he can be motivated to overstate ROCs to receive the buy-out premium (see path *a*). If the supplier has not enough ROCs to cover the obligation, he is motivated to overstate ROCs to avoid the buy-out fee penalties (see path *b*).

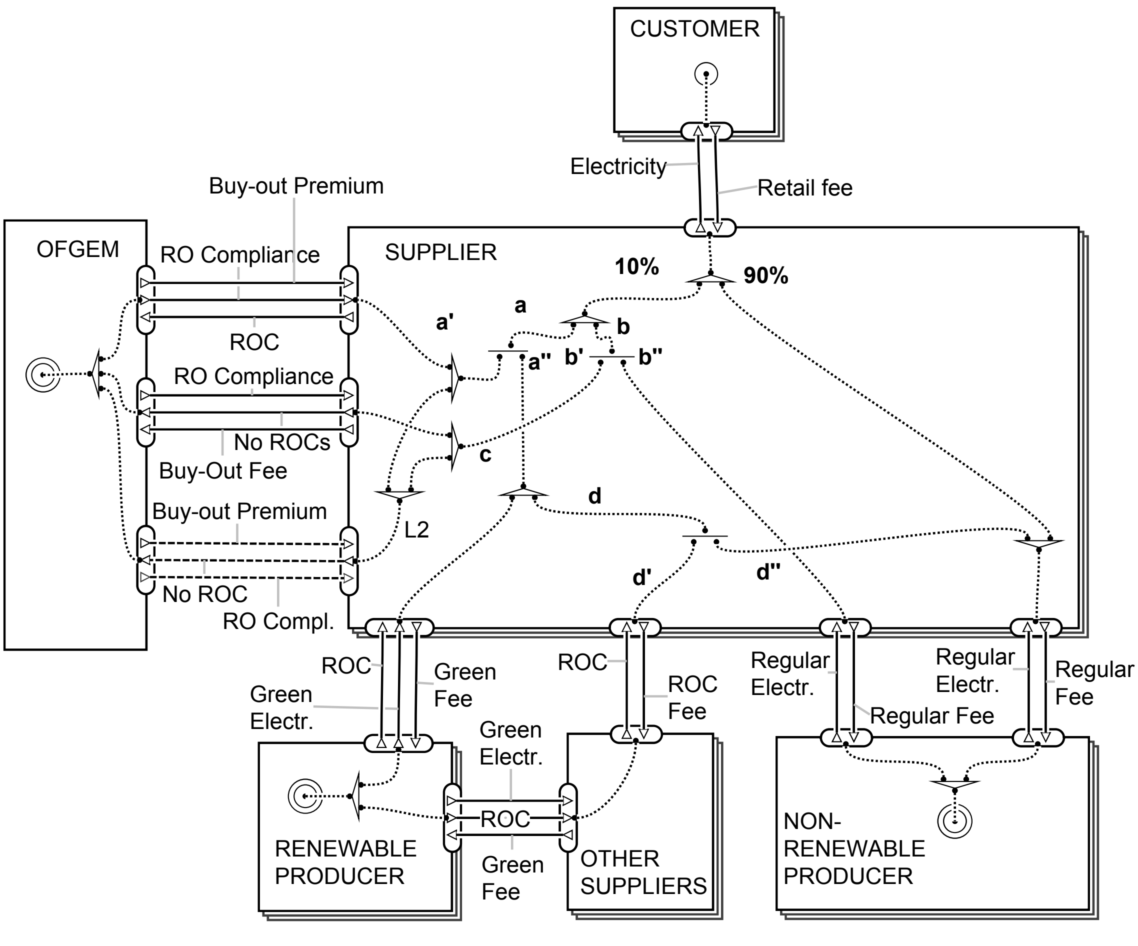


Figure 12.10: A sub-ideal value model with tradable ROCs

12.3.3 Step 3: Select a control pattern

In order to select a pattern for the design of an appropriate control mechanism, we first need to identify the primary actor, the counter actor, primary activities, counter activities, and other elements, as explained in section 11. As in the previous e^3*control* cycle, the Supplier is a counter actor and Ofgem is a primary actor. According to the sub-ideal value model in Figure 12.10, the ideal primary object consists of two value transfers: *RO Compliance* and *BuyOut Fee*. The ideal counter value object is *ROC* and the sub-ideal counter value object is *No ROC*. The sub-ideal counter activity is *Overstate Green Supply*. Note that the counter activity *Buy Green Electricity* can be either ideal or sub-ideal. As we have explained earlier, the supplier has an illegitimate interest in overstating the number of ROCs which does not depend on whether he can cover the RO obligation or not.

As in the previous cycle, the sub-ideal activity *Overstate Green Supply* is a *Sub-ideal Counter Activity*. This type corresponds to the Penalty and Incentive contractual patterns, and the Execution Monitoring and Partner Screening procedural patterns. As in the previous cycle, we are concerned with controlling the sub-ideal behaviour ex-post, and therefore, we limit ourselves to Execution Monitoring, and its variations Penalty and Incentives. Although an additional penalty might be introduced for the supplier who cheats, we concentrate on the process level of the solution. Therefore, we select the pattern Execution Monitoring to be applied as the control for the indicated problem.

12.3.4 Step 4: Apply the Execution Monitoring pattern

Process view. Following the solution given by the Execution Monitoring pattern, we add a new verification activity *Verify ROC* to Ofgem. It verifies the *Present ROCs* activity of the Supplier. That is to say, it checks if a *ROC*, submitted by a supplier, corresponds to a ROC reported by the renewable producer.

In addition, unlike in the ideal model with ROCs in Figure 12.7, the renewable producer not only issues an ROC to the supplier, but also reports the number of issued ROCs to Ofgem. This is modelled with an object *ROC Register*. The ROC register is an electronic, web-based system, supported by Ofgem, which allows generators and suppliers to view the ROCs they hold and to transfer ROCs to other parties. In this way, Ofgem can verify the authenticity of each ROC.

Note that the activity *Report ROCs* is added not because of a pattern's prescription. It is added because the UML language restrains modelling exchange of the object (*ROC Registry* in this case) directly from the AND-join (the black thick bar). We therefore add an activity in between.

In this solution, an *ROC* plays the role of the to-be-verified statement, while the *ROC Register* plays the role of the supporting document. Thus, the renewable producer plays the role of the provider of a supporting document.

After the verification of an ROC, the *Verified ROC* object is used. As in the previous e^3*control* cycle, the *Verified ROC* plays the role of the to-be-verified statement for the verification activity *Verify Compliance*.

The value model does not change and is the same as the ideal value model in Figure 12.8.

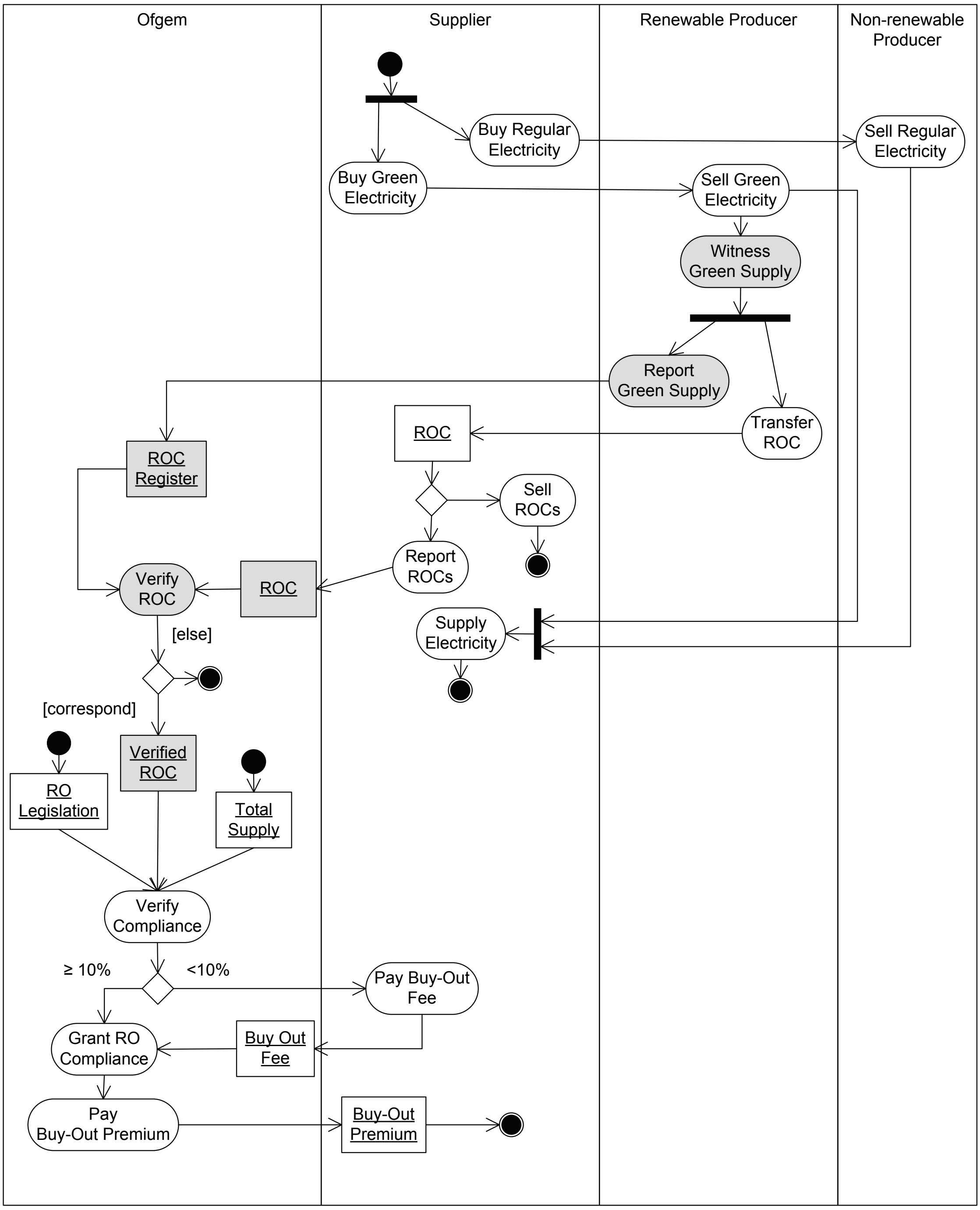


Figure 12.11: Controls in a process model with tradable ROCs

12.4 The size of the buy-out fee and the buy-out premium

This section is devoted to the question of the size of penalties and incentives in the ROC scheme. In the value model in Figure 12.8 the value objects *BuyOut Fee* and *BuyOut Premium* play the roles of a penalty and an incentive correspondingly. To make the penalty and incentive effective, the total value the supplier receives from taking the sub-ideal path must be less than the value he receives from taking the ideal path. The supplier has two ideal paths and one sub-ideal path. The sub-ideal path involves buying electricity from a regular supplier (path *b*). The first ideal path involves getting ROCs by buying green electricity (see path *c*). The second ideal path involves buying ROCs separately and buying regular electricity (path *d*).

12.4.1 Acquiring ROCs together with green electricity

Let us consider the ideal path *c* and the sub-ideal path *b* in Figure 12.8. As explained in Chapter 9, the penalty and incentive for the counter actor should be such that the value he accumulates in the ideal path *c* is larger than the value he accumulates in the sub-ideal path *b*. Then the following condition must hold:

$$-BuyOut\ Fee - Regular\ Fee + Retail\ Fee <$$
$$BuyOut\ Premium - Green\ Fee + Retail\ Fee$$

The left-hand side of the inequality contains the sum[4] of the value objects exchanged by the supplier in the sub-ideal path *b*. The right-hand side of the inequality contains the sum of the value objects exchanged by the supplier of the ideal path *c*.

Note that here and further on we include only *monetary value objects*, in accordance with the *e*^{3}*value* methodology, as described in Chapter 3. Non-monetary objects, such as *Electricity*, come in and out of the supplier and do not change the accumulated net value.

$Retail\ Fee$ on the left and right sides of the inequality can be cancelled. This leaves us with the following inequality:

$$Regular\ Fee + BuyOut\ Fee > Green\ Fee - BuyOut\ Premium \qquad (12.1)$$

This states that to make the incentive and the penalty effective, the payments for green electricity minus the buy-out premiums must be lower for the supplier than the payments for regular electricity and payments to the buy-out fund. Without considering the buy-out premiums, this means that the buy-out fee per MWh should be more than the difference between the green and the regular electricity prices per MWh. Because electricity prices are determined by market mechanisms, the $Regular\ Fee$ and $Green\ Fee$ components in the inequality are constantly changing. To make sure this condition holds, Ofgem must constantly monitor the market and adjust the values of the buy-out fee and buy-out premium if the price of electricity changes. If Ofgem fails on this and allows the buy-out fee per MWh to be substantially lower than the price

[4]Here and further the outgoing value objects are taken with the ‘minus’ sign.

difference, the whole scheme becomes ineffective. The suppliers will prefer to pay the buy-out fee. The condition (12.1) mirrors the reality: As it is stated at Ofgem's web site, the buy-out price is a fixed price per MWh shortfall and is adjusted in line with the Retail Prices Index each year [OFGEM, 2004b].

12.4.2 Buying ROCs separately

Now we consider the second part of the ideal path, where the supplier buys conventional electricity and buys ROCs separately (path *d*). This means that the supplier pays the ***ROC Fee*** plus ***Regular Fee*** instead of paying ***Green Fee***. The following condition should then hold:

$$BuyOut\ Fee - Regular\ Fee + Retail\ Fee <$$
$$BuyOut\ Premium - Regular\ Fee - ROC\ Fee + Retail\ Fee$$

As before, we include only monetary value objects. After cancellation of $Retail\ Fee$ and $Regular\ Fee$ components, the condition for the effectiveness of the buy-out fee and buy-out premium becomes the following:

$$ROC\ Fee < BuyOutPremium + BuyOutFee.$$

This condition states that the price of the ROC per MWh must always be lower than the sum of buy-out fees charged and buy-out premiums paid per MWh. Only if this condition holds, will suppliers prefer ROCs to the penalty. Since the price of ROCs is defined by the market, Ofgem has to adjust the buy-out premium and the buy-out fee according to the price of ROCs.

12.4.3 ROC Fee as a penalty

The underlying idea of the ROCs scheme is to stimulate the production of green electricity to comply with the Kyoto protocol. Only if sufficient green electricity is generated will the UK government, represented by Ofgem, be willing to increase renewable electricity production. The increase of renewable production can be achieved only if sufficient MWh of renewable generation can be supplied. The ROC scheme has mechanisms to motivate building of renewable generation facilities. Let us examine this more closely.

If we assume that the ultimate goal of the UK government, represented by Ofgem, is to increase renewable generation (at least until the level of green energy output prescribed by Kyoto is achieved), then it is fair to state that Ofgem wants to stimulate the suppliers to buy green electricity, as in path *c*, and not to buy regular electricity (with ROCs), as in path *d*. In our terms, Ofgem considers that the path *c* as ideal and the path *d* as sub-ideal. In this respect, ***ROC Fee*** – the price of ROCs – plays the role of a penalty, which the supplier has to pay for not buying green electricity. According to the penalty size condition in section 9.1, to stimulate the suppliers to buy green electricity, the value accumulated when buying green electricity (the path *c*) must be higher compared to the value accumulated when buying ROCs and regular electricity (the path *d*). Thus, the following condition must hold for Figure 12.8:

$$Green\ Fee < ROC\ Fee + Regular\ Fee.$$

If the *ROC Fee* increases, the suppliers will prefer to buy electricity from renewable producers (the path *c*), and not to buy ROCs from other suppliers (the path *d*). Because of the scarcity of renewable production facilities in the UK, only a certain number of suppliers will be able to comply with their required 10% of green supply by buying electricity from renewable producers in path *c*. As soon as path *c* is exhausted, the suppliers will need to either buy ROCs on the ROC market (path *d*) or pay the buy-out fee (path *b*). If the condition of the previous section $ROC\ Fee < BuyOutPremium + BuyOutFee$ holds, suppliers will prefer to buy ROCs rather than to pay the buy-out fee. As a result, the demand for ROCs will grow, thereby pushing up their price *ROC Fee*. In such a situation, the only way to reduce the *ROC Fee* and to avoid paying the buy-out fee is to build more renewable generation facilities and increase the supply of green electricity in path *c*. This is what the UK government hopes the effect of the RO regulation will be.

12.5 Discussion

We have demonstrated how the combination of the Penalty and Incentive contractual patterns and the Execution Monitoring procedural pattern can explain controls in the ROC scheme. The models in Figure 12.8 and Figure 12.7 represent a part of the actual scenarios of ROCs exchange. The correspondence with reality was ensured by comparing the results of the patterns with our knowledge about the ROC scheme obtained from documentary sources and the BusMod project (see Research Context section).

The pattern also allows us to reason about the size of the penalties and incentives which have been introduced. As was demonstrated, an ROC plays the role of the penalty in stimulating the building of new renewable generation sources.

As reported in Ofgem's in their Annual Report of 2004-2005[5], the ROC scheme has had an effect. There has been a growth in the amount of electricity supplied from renewable sources. The report shows that in 2004-2005 (the third year of the scheme) more than 10.8 million ROCs were issued compared with 7.5 million in 2003-2004 and 5.5 million in 2002-2003. There were also a total of 788 generating stations participating in the scheme at the end of 2005 up from 505 at the end of the first year.

12.6 Lessons Learned

In the lessons learned we reflect on our expectations about value modelling for controls and compare them to the results of this case study.

[5] [OFGEM, 2006]

Lesson 1. Controls as commercial services with added value. This case study does not contain such a variety of control services as the health care case study in Chapter 5. There is only one control service: The renewable producer executes a witnessing of the supply on behalf of Ofgem.

What is remarkable with regards to value models, is that the principle of reciprocity of e^3*value* forces us to think about what should be exchanged *in return* for such a service. In this case, we assume that the service is not a commercial service: Ofgem does not pay the producer for it. Empowered by the status of a government agency, Ofgem obliges the producer to submit this information. In value models, this service could be modelled by a value transaction, where one value object indicates the service delivered by the producer to Ofgem and the reciprocal value object *Compliance* indicates that the producer complies with the regulation.

From the process perspective, the control service has been shaped as a result of delegation of the witnessing control activity by a primary actor to a trusted actor. So, at a process level, the controls service corresponds to a delegated activity.

Although the described service is not commercial, the existence of such a service as such demonstrates that the control can be seen as commercial services. This justifies the use of the value modelling in the design of controls, as suggested in this thesis.

Lesson 2. Value aspects of control instruments. In this case study, we have an extensive variety of control instruments: penalties, incentives and evidence documents.

As far as penalties are concerned, we have encountered the *buy-out fee* that represents a penalty paid by the supplier for not complying with the renewable obligation. As for incentives, the *buy-out premium* payment has been described, which is a reward to the suppliers who complied with the renewable obligation. Remarkable in this case is that the premiums were paid from the collected buy-out penalties.

The core evidence document in the case study is the Renewable Obligation certificate or ROC, which was introduced to keep track of renewable energy supplied by each supplier. The evidence document ROC has been modelled as a value object for the following reasons: (1) the ROC can be traded and (2) the ROC is of value to Ofgem, who needs trustworthy evidence of green supply.

The ROC is even more interesting, because it is a value object that has a price and this price plays the role of another penalty. The price of the ROC is a kind of a dynamic penalty that has a goal to stimulate the suppliers either to buy green electricity or to build new renewable generation facilities, if there is not sufficient green production to cover everyone's need.

Lesson 3. The value viewpoint is an abstract rationale for procedural aspects of controls. This case study demonstrates that modelling controls with value models is possible. The concepts of e^3*control*, such as sub-ideal value object, sub-ideal transfer and sub-ideal path, are extensive enough to represent the discovered control problems. Namely, by using the OR- and AND-forks, it is possible to model various kinds of ideal and sub-ideal choices the supplier can make. Every control problem identified in this case could be represented by these choices and corresponding sub-ideal value objects. For each value object we could identify a set of operational activities and objects in the process model.

The value perspective proves to be useful, since it is impossible to describe the ROC scheme by taking the process perspective only. The scheme requires the implementation of both value-related elements (the penalties *Buy-Out Fee* and *ROC Fee*, and the incentive *Buy-Out Premium*) as well as process-related elements (*Verification* and *Witnessing* activities and the evidence document *ROC*). Without analysing ROCs at value level it would be unclear, other than providing evidence, what the rationale behind the ROCs is.

Lesson 4. Patterns. The controls of the study could be described using the Penalty, Incentive and Execution Monitoring patterns. In addition, both delegation patterns have been used.

In general, we have not found anything of a serious nature that could not be expressed by the patterns. Only at one point in section 12.3.4 did we have to add an activity *Report ROCs*, which is not described by the applied pattern. This was necessary in order to obtain a correct UML model and to avoid modelling a transfer of an object from an AND-joint and an activity directly.

We have also managed to create models which comply with the e^3*value* ontology is the sense that they do not contain loops in the dependency paths. Neither do the ideal value models contain one-way value transfers.

12.7 Summary

In this chapter, we have presented patterns in the case of Renewable Obligation (RO) regulations in the UK. We have demonstrated how the combination of the contractual patterns Penalty and Incentive and the procedural pattern Execution Monitoring can explain the market of Renewable Obligation Certificates (ROCs) and processes behind it. ROCs are tradable securities representing evidence of generation of renewable electricity in Mega-Watt/hours.

We have also demonstrated the process of selection and application of the patterns, as described in Chapter 11. The models resulting from the application of the patterns closely match a real situation with the RO regulation, which demonstrates that the patterns are usable for designing solutions to real-life control problems.

Chapter 13

Beer export

In this chapter we proceed our evaluation of the relevance, consistency, and effectiveness of our library of control patterns. We introduce a case study, which concerns the redesign of Pan-European customs procedures for collecting excise duties. The focus will be on the export of excised goods from the Netherlands to other countries within the EU, in particular, the UK and Poland.

13.1 Research Context

This case study is part of an integrated project called ITAIDE (Nr.027829), which is funded by the 6th Framework IST Programme of the European Commission (see www.itaide.org). The ITAIDE project was motivated by a challenge that European governments are facing at the moment; that of solving the dilemma of increasing the security and control of international trade, while at the same time reducing the administrative overhead carried by commercial and public administration organizations. Part of the ITAIDE project is the Beer Living Lab (BeerLL)) which was set up to redesign EU customs procedures. It focuses on procedures for shipments of beer from the Netherlands to destinations outside the EU (export) and within the EU (intra-community supplies). Collaboration between one of the worlds largest beer producers [1], the Dutch Tax and Customs Administration, two very large technology providers and various universities aims to demonstrate that trade facilitation, reduced administrative burden for supply chain partners and improved control and security are not necessarily contradictory efforts and can actually coexist.

The scope of this case study differs from that of the BeerLL in the sense that we consider problems associated with untrustworthy beer producers. The BeerLL considers the beer producers to be reliable and trusted parties. The contribution of this case study to the BeerLL is an analysis of the case using control patterns.

The specificity of such a research context is that part of the data collection was accomplished through participation in joint workshops with domain experts, such as customs officers and managers of a beer producing company, who were BeerLL members. The goal of the joint

[1]The beer company is anonymous, but information is available upon request with the author.

workshops was to come up with a redesign of current customs procedures. This means that the whole research setting has elements of action research, since not only could the researchers observe the decisions made by the domain experts, but they could also contribute to the discussions and decision making. Moreover, the ideas generated in the workshops as well as the results presented in this case study were implemented in the real-life demonstrator, which consisted of about ten containers shipping goods to various countries within and outside of the EU.

In addition to the redesign workshops, data was collected by conducting multiple interviews with the domain experts of the participating companies as well as with external domain experts. Data collection was carried out by the university researchers of the ITAIDE project, including B. Rukanova, Z. Baida, J. Liu, Y.-H. Tan [Baida et al., 2007]. In addition to data collected by others for the ITAIDE project, the author also collected own data for the case study. The author did this by participating in five project meetings and sessions of the ITAIDE project and conducted one extensive interview with two representatives of Customs NL. The interview specifically focused on control problems in export procedures.

The e^3*control* and process models of the case study were many times presented in Power Point slides during redesign workshops and discussed extensively by the domain experts. Furthermore, the models of the redesigned procedures were verified in a workshop with the domain experts. The domain experts confirmed the correctness of the models.

13.2 Excise collection within the EU

We now go on to describe the current excise collection procedure. More details can be found in [Baida et al., 2007] and [Rukanova et al., 2006].

When excise goods like beer and cigarettes are sold, the seller must pay a special tax called *excise*. Within the EU, the general principle is that excise is only paid in the country in which the excise good is consumed. Currently, excise tariffs differ among different EU countries. Hence, if a beer producer in the Netherlands (BeerCo NL) exports beer to a retailer in the UK and the beer is consequently sold to UK consumers, then the excise duty has to be paid in the UK. In this case, the BeerCo NL does not pay excise in the Netherlands.

Since there are officially no borders between the EU member states, sellers are not obliged to report exported goods to the customs of the country of export. So, BeerCO NL is not obliged to report to Dutch Tax and Customs Administration (Customs NL). The customs agencies cannot therefore physically follow the goods exported from the country of their jurisdiction for exports within the EU.

Clearly, the excise free export is only acceptable for Customs NL, if there is *trustworthy evidence* that BeerCo NL shipped the goods abroad. Since there are no border controls, the customs have to rely on other evidence. Currently, the core evidence of export in the EU is the paper Administrative Accompanying Document (AAD). The AAD contains information about goods in the associated container or truck. According to the law, BeerCo NL should receive *excise exemption* upon presenting the AAD stamped by Customs UK. By stamping the AAD, Customs UK *testify* that the goods stated in the AAD have arrived to the UK. The second purpose of the AAD is as a means to identify the cargo in case of a physical cargo inspection en route.

Verification of every imported container or truck is very labor-intensive for Customs UK. Therefore, customs delegates the actual verification of the goods to a special retailer with the status of *Excise Warehouse (EW)*. To receive the status of EW, the internal controls of a retailer have to be *approved* by audits carried out by Customs UK. If the auditing process by Customs UK is satisfactory, the retailer is granted the EW license, and he is allowed him to operate with excise-free exports. Excise-free export is only permitted if done through a EW retailer.

The procedure of stamping an AAD is as follows. The AAD accompanies the beer from the Netherlands to the UK and is stamped by the Retailer UK (with EW status) as proof that the goods have arrived in the UK. Further Customs UK also stamps the AAD. The UK customs agency sends the stamped AAD back to the UK retailer who will forward it to BeerCo NL.

In the Netherlands, Customs NL periodically checks BeerCo NL's excise declarations. The excise exemption is given by default for all the beer that BeerCo NL declares as being sold outside the Netherlands. The fact of export is verified afterwards by comparing excise declarations with AADs. As transferring paper-based AADs can take weeks or months, the actual verification is done several months later. In practice, verification is often not done at all because it is labour intensive. In fact, the BeerCo NL only submits AADs upon the request of the Customs NL.

The controls based on the paper AAD lead to an administrative burden and possible excise errors or even fraud, as was indicated by the domain experts. For example, the AAD is a paper document that can be lost, and this happens frequently according to the domain experts. In addition, the AAD can be tampered with. According to the EU Commission [Brussels, 2006], excise fraud for alcohol in the EU amounts to 1.5 billion yearly, approximately 8% of the total excise duties receipts on alcoholic beverages. The EU, therefore, intends to introduce e-Customs for excise goods, which means that paper-based control procedures, such as the AAD procedure, will be replaced by electronic ones.

13.3 Case study structure

The goal of this case study is two-fold. Firstly, we analyze the control problems that underlie the AAD procedure and what controls constitute the procedure. We are also interested in why the AAD procedure does not provide sufficient controls. To answer the last question we compared the AAD procedure with the solution suggested by the e^3*control* patterns and identified the control weaknesses of the AAD procedure compared to the patterns.

The second goal of the case study is to suggest a scenario on how the AAD procedure can be improved using innovative information technology. In this case, the e^3*control* patterns are used to verify whether the new solution using the information technology will provide better controls compared to the AAD procedure.

We have identified two common illegitimate manners of avoiding excise payments in the Netherlands[2]. The first is when the beer producer in the Netherlands BeerCo NL overstates the ex-

[2]Even more control problems exist in the customs procedures, but in the case study we focus only on these two

ported goods and understates the goods sold in the Netherlands. To carry this out, the AAD must be forged. According to domain experts, BeerCo NL is able to forge an AAD and overstate the amount of exported beer using the existing paper-based procedure. We consider this problem in the Tampered AAD scenario.

The second way of avoiding excise payments is referred to by domain experts as ‘virtual beer’. The term ‘virtual’ refers to the export of beer that does not take place physically, but only ‘on paper’. In this case, the administration of importers states that beer is shipped and sold outside the Netherlands, while in reality it is sold in the Netherlands on the ‘black’ market. In this way, excises are paid in the country of ‘virtual’ export, which has lower excise tariffs. This scenario is even more attractive to fraudsters when they avoid excises in the country of ‘virtual’ export. Avoiding these excises might be easier when the ‘virtual’ export is done to a country with high corruption.

The rest of this chapter is structured as follows. In section 13.4 we describe the first e^3*control* cycle, in which we consider the Tampered AAD scenario, which describes the current AAD procedure. After analyzing the problem, we suggest a solution, resulting in the value and process control models, which are the basis of the patterns. By comparing the current AAD procedure with these models, the weaknesses in the AAD procedure are pin-pointed in more detail.

Furthermore, in section 13.5, we analyze to what extent the current initiatives to replace the AAD procedure would really improve the situation. In section 13.6 we describe the second e^3*control* cycle, in which we consider the Virtual Beer scenario. In this section, we extend the control solution models created in the first cycle to solve the ‘virtual’ beer problem. Amongst other things, this solution explains why EW licensing is needed. Finally in section 13.7 we suggest a new improved procedure that can be implemented using new TREC technology.

13.4 Tampered AAD scenario

In this section we consider the control problem when BeerCo NL avoids excise payments in the Netherlands by faking the AAD so that it overstates the exported goods and understates the goods sold in the Netherlands. We suggest improvements in the AAD procedure to combat this risk. We start the analysis by modelling the AAD procedure using ideal value and process models. We proceed with modelling the control problems in the AAD procedure with the sub-ideal value and process models. Finally, we suggest a solution using patterns. As we shall see, the AAD procedure already contains some elements of the Penalty pattern.

13.4.1 Step 1: Design ideal situation

Firstly, we model an ideal situation for the AAD procedure. We assume that no party commits fraud by tampering with AADs and that there are no missing AADs.

To summarize, the following actors are involved in this study: (1) BeerCo NL, a large Dutch beer producer; (2) BeerCo UK, the UK branch of BeerCo NL, functions as an intermediary

problems.

between BeerCo NL and retailers in the UK; (3) Customs NL, the Dutch customs; (4) Customs UK: the British customs; (5) Retailer UK with EW, an EW retailer of beer in the UK; (6) Consumer UK, an actor who combines two roles, representing supermarkets in the UK that buy Dutch beer from Retailer UK and final consumers of the beer in the supermarkets; (7) Retailer NL, a wholesale warehouse of beer in the Netherlands; (8) Consumer NL, an actor representing end consumers and supermarkets in the Netherlands.

Value view. The ideal value model in Figure 13.1 shows the BeerCo NL who either sells beer within the Netherlands or exports it to the UK. Since the beer is sold in the two countries - the UK and the Netherlands- the dependency path has two consumer needs: one need of the Consumer NL and the other need of the Consumer UK.

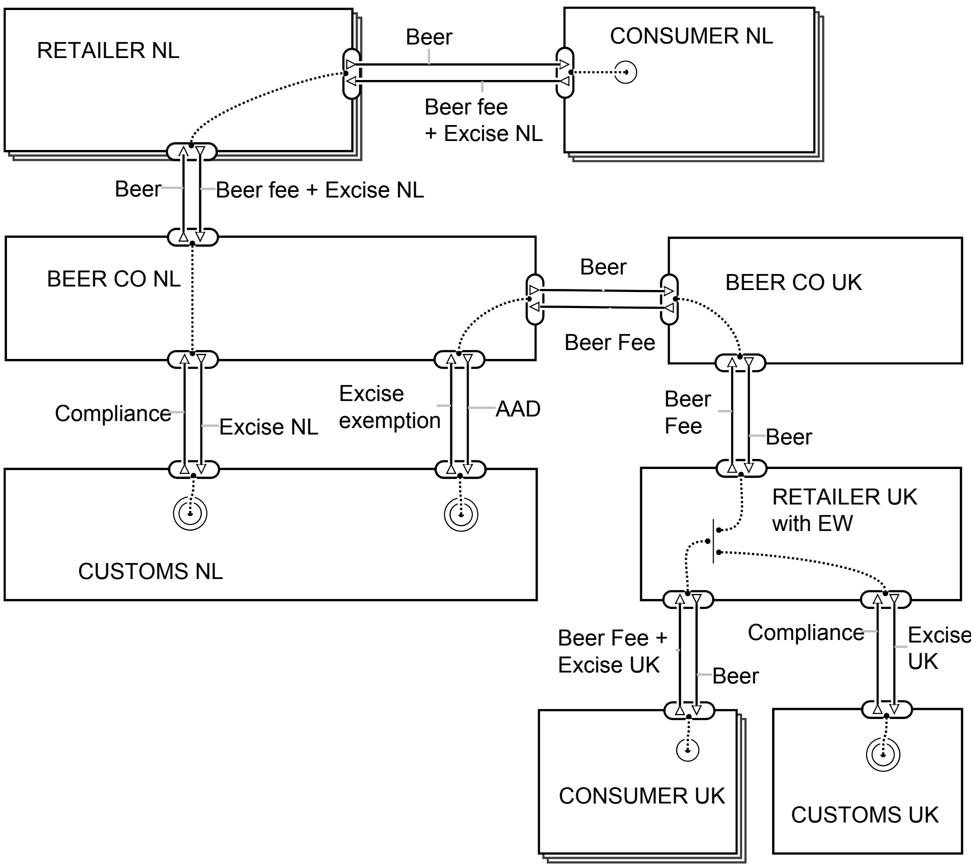


Figure 13.1: Ideal value model of the beer sales

Consumer NL buys beer from Retailer NL. In return, Consumer NL pays a fee, which includes excises according to the Dutch excise tariff. This is modelled with the value object *Beer*, flowing to Consumers NL from Retailer NL, and the value object *Beer Fee + Excise NL*, flowing in return. BeerCo NL, who sells beer to Retailer NL, transfers *Excise NL* to Customs NL and in return receives the value object *Compliance*, which indicates that by paying excises BeerCo NL complies with the excise law.

Similarly, Consumer UK buys beer from Retailer UK. In return, Consumer UK pays a $Beer\ Fee$

and excises for the beer according to the UK excise tariff, $Excise\ UK$. Retailer UK buys beer from BeerCo UK, who in its turn buys it from BeerCo NL.

Excise UK is paid to Customs UK by Retailer UK. BeerCo UK and BeerCo NL trade beer without excises. Therefore, in the Netherlands, BeerCo NL gets *Excise Exemption* by presenting the *AAD*.

Additional actors are Carrier NL who ships beer to the Netherlands and Carrier UK who ships beer to the UK. We omit the carriers in the value models for the sake of simplicity of visualization. We assume that the carriers work on behalf of BeerCo NL.

Process view. Figure 13.2 shows a process model that corresponds to the ideal value model. In this model, we only focus on export to the UK.

This process was built based on a model in [Rukanova et al., 2006], p.17, built within the ITAIDE project; the model was also approved by ITAIDE domain experts. In this case, not all the process details are shown, but only those which reflect elements important in the excise declaration process.

The process starts with Retailer UK ordering beer from BeerCo UK. BeerCo UK orders beer from BeerCo NL. BeerCo NL ships beer to Retailer UK with EW status. According to the regulation, BeerCo NL is eligible for excise exemption in the Netherlands when the goods are exported out of the country. This happens when the carrier *crosses the border* of the Netherlands. So, we model BeerCo NL who performs a *Ship Beer* activity and the Carrier who performs a *Cross Border* activity on behalf of BeerCo NL. Retailer UK performs a *Receive Beer* activity.

After receiving the beer, Retailer UK stamps the AAD document and transfers it to Customs UK. Customs UK also stamps the AAD and transfers it back to Retailer UK. Retailer UK forwards the AAD to the Carrier. Finally, the Carrier delivers the document to BeerCo NL.

Parallel to this, and after the beer is shipped, BeerCo NL claims excise exemption by providing an *Excise Declaration* to Customs NL. The *Excise Declaration* contains information about exported goods and goods sold inside the Netherlands.

Although in practice excise exemption is granted to BeerCo NL based on the *Excise Declaration*, the declaration has no legal power to be a reason for excise exemption. Only the AAD has this legal power, which means that BeerCo NL has always has to back up the requested excise exemptions with AADs. In reality the AAD is not checked 100% (e.g. because it is labor-intensive). We therefore model that BeerCo NL presents the AAD to Customs NL only if Customs NL asks for it (see the choice [check] and [do not check] at Customs NL). If verification of the AAD shows that the export is eligible for exemption, Customs NL grants the excise exemption to BeerCo NL , see activity *Grant Excise Exemption*. If the AAD check is not carried out, excise exemption is granted without the AAD, and is based solely on the *Excise Declaration*.

If the check is done, the primary actor Customs NL executes the verification activity *Verify Export*, by which the information in the AAD is reconciled against the *Excise Law*. The excise law described what valid export evidence actually is. For example, an AAD that has not been stamped by Customs UK is not valid export evidence.

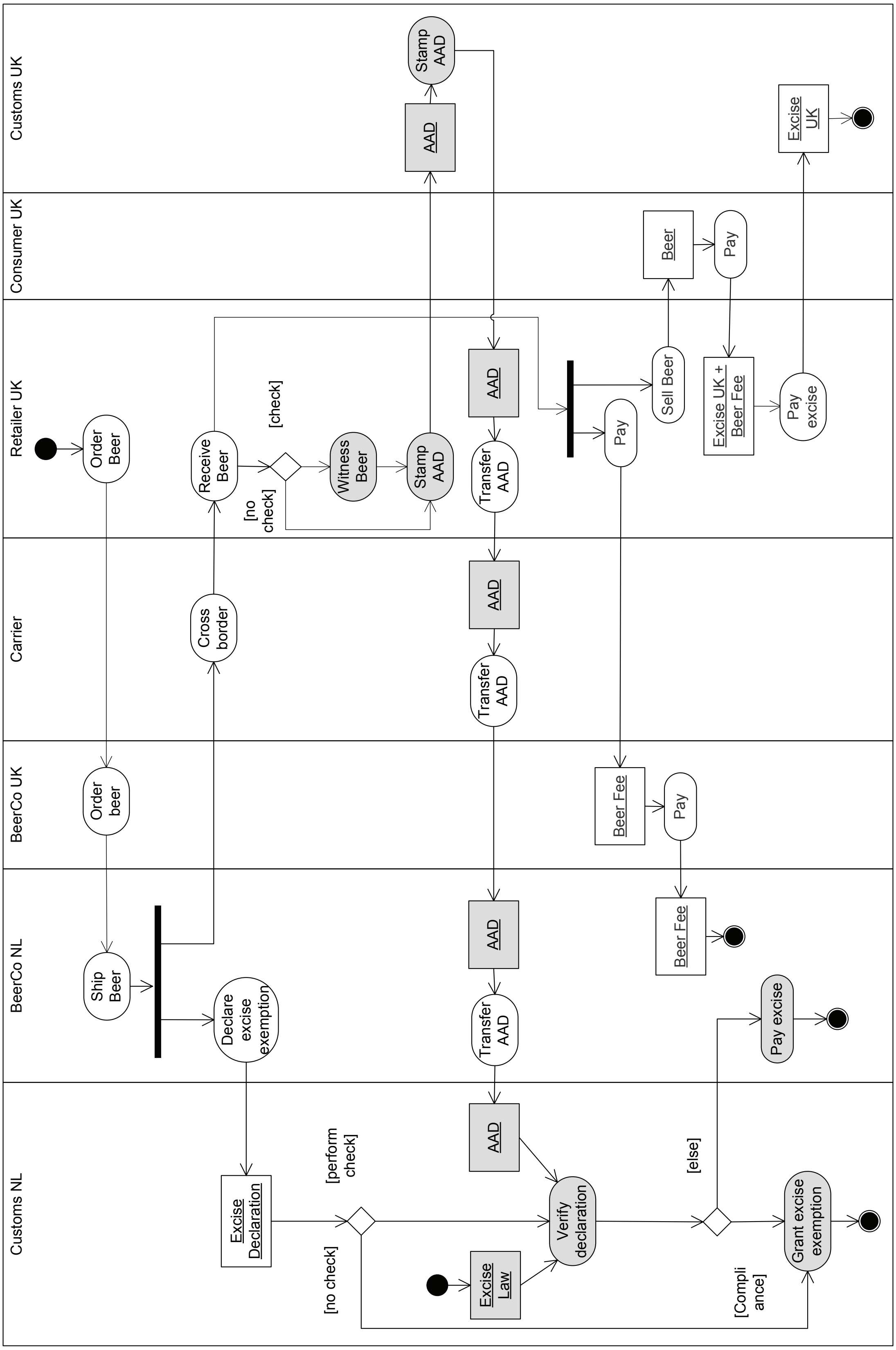


Figure 13.2: The ideal process model of the beer sales

13.4.2 Step 2: Design sub-ideal situation

The control problem in the Tampered AAD scenario is that BeerCo NL can overstate the amount of exported beer and claim excise exemption for beer that is actually sold within the Netherlands.

Value view. Figure 13.3 shows the sub-ideal value model. If the beer is exported to the UK, BeerCo NL has a choice of submitting a correct *AAD* or of overstating the scale of export by forging the AAD. This choice is modelled with the *OR-1* fork. The forging activity results in a sub-ideal value transfer *Fake AAD*. In both cases, *Excise Exemption* is granted. However, if a *Fake AAD* is presented, then this excise exemption is sub-ideal, since it is granted for the export that did not occur. The exchange of *Fake AAD* is marked with the liability token **L1** at BeerCo NL.

In this model we assume that BeerCo NL overstates the exported beer amount, but does not overstate the total beer sales. This is realistic, since to overstate total sales, BeerCo NL would have to make other administrative manipulations. In addition, the higher reported sales will result in higher income taxes, which makes the attractiveness of the whole excise fraud questionable for BeerCo NL. Therefore, when overstating export, BeerCo NL will always understate sales in the Netherlands[3].

This is modelled with the OR-forks and AND-join. If the BeerCo NL overstates the amount of exported beer at the *OR-1* fork, then he understates the amount non-exported beer at the *OR-2* fork. Both forks lead to the sub-ideal exchange through the AND-join.

Process view. Figure 13.4 represents the sub-ideal process model that corresponds to the sub-ideal value model in Figure 13.3. In the process model, we show only the sales of beer to the UK and assume that BeerCo NL overstates these sales. The beer that is declared for excise exemption is illegally sold in the Netherlands to Retailer NL, not in the UK to the Retailer UK. As a result, BeerCo NL's carrier does not execute the *Cross Border* activity. Furthermore, BeerCo NL overstates the excises by faking the AAD, see activity *Present Fake AAD*. Finally, the excise declaration is also incorrect, since it has to correspond with the AAD. The activity *Grant Legal Compliance*, executed upon the fake AAD is also sub-ideal.

13.4.3 Step 3: Select a control pattern

To solve the control problem, we should introduce an extra control mechanism. In this step we select a pattern, which is appropriate to describe the control problem of the Tampered AAD scenario.

[3]For example, if twenty-five thousand bottles of beer were exported to the UK, seventy-five thousand bottles were sold in the Netherlands, but the fake AAD states that fifty thousand bottles were exported, then the excise declaration would state that fifty thousand bottles were exported and fifty (not seventy-five!) thousand bottles were sold in the Netherlands

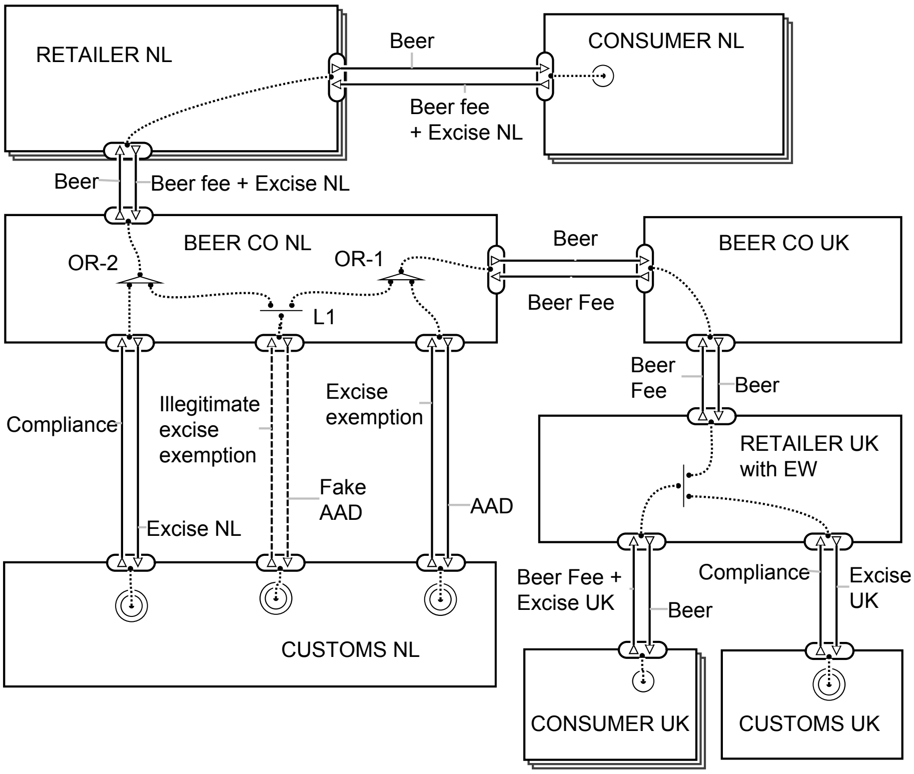


Figure 13.3: Tampered AAD: a sub-ideal value model

Identify primary actor, counter actor, primary object and counter object, ideal and sub-ideal primary and counter activities

To find a pattern for a problem, we first need to identify a primary actor, counter actor, primary value object and counter value object. Then we identify primary and counter activities. For ideal activities, we use the ideal process model in Figure 13.2. For ideal activities, we use the sub-ideal process model in Figure 13.4.

Primary Actor. The primary actor in this case is Customs NL. This is the party that represents the government interested in bona fide excise payments. The control of the excise process is the primary responsibility of the customs agencies. Since the problem concerns avoiding the payment of excises in the Netherlands, Customs NL, and not Customs UK, is the party responsible for the control.

Counter Actor. The counter actor is BeerCo NL, who is considered to behave sub-ideally and has the liability token **L1** in the sub-ideal value model in Figure 13.3. BeerCo NL plays the role of *agent* in the principal-agent relationship with Customs NL: he has more information about his actual beer sales and can hide the information from Customs NL.

Sub-ideal Counter Value Object. The result of the sub-ideal behavior of BeerCo NL is the sub-ideal value object *Fake AAD* in the sub-ideal path marked with **L1**. *Fake AAD* is a sub-ideal counter value object.

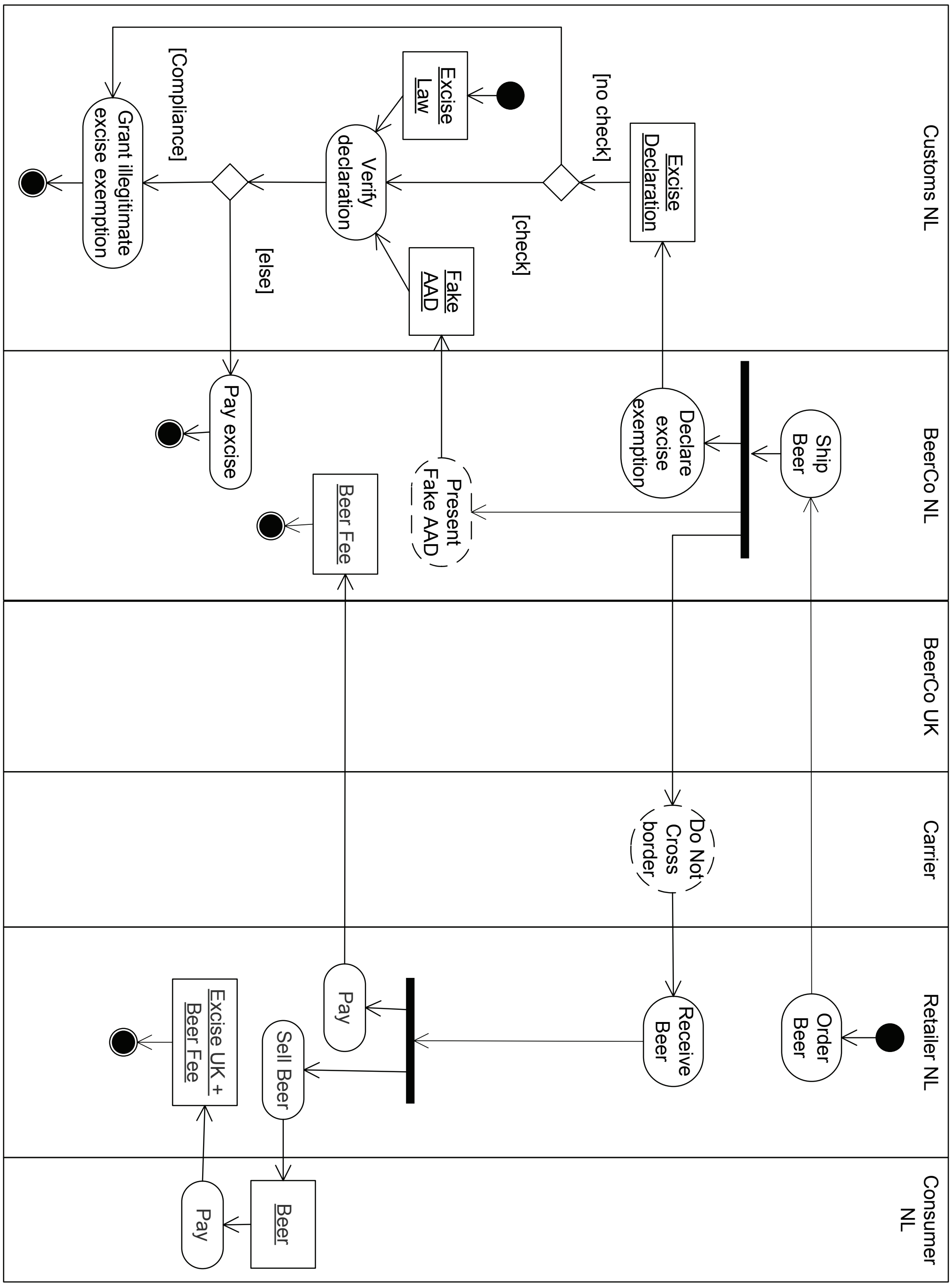


Figure 13.4: Tampered AAD: A sub-ideal process model

Ideal Counter Value Object. In this ideal path of the export to the UK, the value object *Fake AAD* is replaced by the value object *AAD*. Thus, *AAD* is the ideal counter value object.

Sub-ideal Primary Value Object. In Figure 13.3, the sub-ideal value object *Illegitimate Excise Exemption* transferred by the primary actor Customs NL in return for *Fake AAD* is the sub-ideal primary value object.

Ideal Primary Value Object. The ideal primary value object is *Excise Exemption*, exchanged between BeerCo NL and Customs NL in return for the ideal counter value object *AAD*.

Primary Activities. Primary activities are (1) executed by the primary actor Customs NL or his trusted party and (2) result in the value transfer of the primary value object *Excise Exemption*. This makes the activity *Grant Excise Exemption* in Figure 13.2 to be the primary activity.

Sub-ideal Primary Activities. The activity *Grant Illegitimate Excise Exemption* in Figure 13.4 is the sub-ideal primary activity.

Counter Activities. Counter activities are (1) executed by the counter actor BeerCo NL or his trusted party and (2) result in the value transfer of the counter value object *AAD*. As can be seen from the ideal process model, in order to transfer *AAD*, BeerCo NL must first exports the goods, and then, present the AAD. The activities *Cross Border* and *Present AAD* are therefore counter activities. *Cross Border* is not the obvious match with the counter value object *AAD*. However, this is also a counter activity, since in the ideal situation presenting the AAD always occurs when crossing the border occurs. Although the *Cross Border* activity is executed by the Carrier, not crossing the border is a responsibility of BeerCo NL, since we assume that the Carrier is a sub-contractor of BeerCo NL.

Sub-ideal Counter Activities. There are several operational activities that correspond to the transfer of the sub-ideal counter value object *Fake AAD* (see Figure13.4). Firstly, the value transfer *Fake AAD* corresponds to the activity *Present Fake AAD*. Secondly, every time the fake AAD is issued, the beer does not cross the border. That is why the *Cross Border* activity does not take place. So, *Present Fake AAD* and *Do Not Cross Border* are sub-ideal counter activities (see Figure13.4).

As a result, we identify two sub-ideal activities: *Do Not Cross Border* and *Present Fake AAD*. In the sub-ideal process model they are marked with dashed lines.

To select a pattern, we identify the *type* of sub-ideal activities *Do Not Cross Border* and *Present Fake AAD*. In Chapter 7, we distinguish the following types of sub-ideal activities: Sub-ideal Counter, Emerging Sub-ideal, Deny Commitment and Deny Execution (see Chapter 11 for more details). Both *Do Not Cross Border* and *Present Fake AAD* activities are of the type Sub-ideal Counter. *Do Not Cross Border* corresponds to the ideal counter activity *Cross Border*. *Present Fake AAD* corresponds to the ideal counter activity *Present AAD*.

Step 3.2. Match the sub-ideal activity to a pattern

As was explained in Chapter 11, a **Sub-ideal Counter** corresponds to the Partner Screening, Execution Monitoring, Penalty, and Incentive patterns. We can apply each of these patterns to both counter activities *Present AAD* and *Cross Border*.

However, we stop the pattern selection process at this stage, since our goal is not to design a new control but to see if the AAD procedure can be improved. Therefore, we examine whether the AAD procedure corresponds to any of the suggested patterns. In fact, the AAD procedure is nothing more than an ex-post verification of the *Cross Border* activity. As we can see from Figure 13.2, the AAD procedure contains elements of the Execution Monitoring pattern. These elements include:

Verify activity. The primary actor Customs NL executes the verification activity *Verify Export* and grants excise exemption based on verification of the AAD with the excise legislation. For example, if the legislation states that excise exemption is only granted for exports to the EU countries, then an AAD coming from the US will not result in excise exemption.

To-be-verified and Supporting Statements. In the *Verify Export* activity, the AAD represents the *To-be-verified Statement* and the *Excise Law* represents the *Supporting Statement*.

Witness activity. The activity *Witness Export* which is the witnessing of the *Cross Border* activity is executed in the AAD procedure by Retailer UK with EW status. Retailer UK sees the delivered beer and testifies to it by stamping the AAD.

Testifying Chain. According to the Execution Monitoring pattern, the *Cross Border* activity should be witnessed by the primary actor Customs NL. However, the Dutch customs authority is not able to witness when the beer containers or trucks leave the country. Since there are officially no borders between the EU member states, exporters are not obliged to report exports to customs. Therefore, the customs agencies cannot physically follow goods exported from the country of their jurisdiction within the EU. Thus, Customs NL does the verification of export based on the AAD, which was stamped by Customs UK. In our terms, Customs NL *delegates* the *Witness Export* activity to Customs UK. Furthermore, as explained earlier, the actual witnessing is not done by Customs UK, but by Retailer UK with EW. Customs UK only stamps the AAD, already stamped previously by Retailer UK with EW. In our terms, we say that Customs UK also *delegates* the witnessing activity to Retailer UK.

This AAD procedure can therefore be described as a combination of the Execution Monitoring pattern with the Testifying Chain delegation pattern, described in section 10.3. According to the Testifying Chain pattern, Customs UK plays the role of the *Primary Actor*, to whom Customs NL delegates the control activity *Witness Export*. Retailer NL plays the role of the *Trusted Actor* of Customs UK. The *Stamp AAD* activities are *Testify* activities of the Testifying Chain pattern, and the AADs that come out of the *Stamp AAD* activities, are the *Testifying Statements* of the Testifying Chain pattern. For more details on the Testifying Chain pattern see section 10.3.

The design of an improved AAD procedure can be seen as a task for ensuring that the existing model complies with the Execution Monitoring pattern and the delegation pattern Testifying Chain. Some elements of the patterns are already in place in the AAD procedure. However, we must ensure that they are applied correctly, according to the patterns' control principles.

In addition, we would like to punish BeerCo NL for faking the AAD. This can be done by applying the Penalty pattern. The Penalty pattern, which is a variation of the pattern Execution Monitoring, can therefore reuse the existing elements of the AAD procedure.

13.4.4 Step 4: Apply the Penalty pattern

In this section we check whether the current AAD procedure complies with the selected Penalty control pattern and, if not, we correct the discrepancies. As a result, we suggest an improved procedure for excise exemption. In Table 13.1 we list all the control principles of the Penalty pattern. The principles have already been adjusted for delegation. In the last column of the table, we note whether the principle has been observed or not in the AAD procedure in Figure 13.2. In square brackets we indicate how the elements of the principles are instantiated in the case. The original control principles can be found in Appendix A.

Table 13.1: Verification of compliance of the AAD procedure in Figure 13.2 with the Penalty pattern

ID.	**Control principles**	**Y/N**
	Activities-Activities principles	
V-Ia	Witness [Witness Export] activity must exist	Y/N
V-II	Verify [Verify Export] activity must exist	Y/N
V-IIIa	Witness [Witness Export] activity must be executed at the same time as Counter [Cross Border] activity	N
V-IVa	Verify [Verify Export] activity must follow Witness activity	Y
PE-I	Verify [Verify Export] activity must precede Primary activity [Grant Legal Compliance] and [Pay Penalty] activity	N
	Activities-Objects principles	
V-V	Supporting statement [Excise Law] must be directly transferred to Verify [Verify Export] activity from a source that generates it	Y
V-VIa	To-be-verified statement [AAD] must be directly transferred from Witness [Witness Export] activity to Verify [Verify Export] activity	N
	Activities-Actors principles	
V-VIIa	An actor executing Witness activity [Retailer UK] must be independent and socially detached from the actor executing Counter activity [Carrier]	Y
V-VIII	An actor executing Verify activity [Customs NL] must be independent and socially detached from the actor executing Counter activity [Carrier]	Y

V-IX	Supporting statement [Excise Law] must generated by an actor independent and socially detached from the actor executing Counter activity [Carrier]	Y

As Table 13.1 shows, not all the control principles of the Penalty pattern are obeyed. Firstly, the activity *Pay Penalty* of the pattern is instantiated in the AAD procedure, as described in the ideal value model[4]. This is indicated by the principle *PE-I* of Table 13.1.

Secondly, the AAD is not transferred *directly* to the primary actor Customs NL from Retailer UK, the actor who does the witnessing. The AAD is transferred first to the Carrier, then to BeerCo NL, and then to Customs NL. This violates the pattern, which requires the to-be-verified document AAD to be transferred directly to Customs NL, as indicated by the principle *V-VIa* of Table 13.1. Only if the AAD cannot be forged (which is not the case here), is an indirect transfer acceptable. According to the domain experts, this diversion from the pattern indicates a real and existing control problem.

Thirdly, because of a lack of resources, Customs NL does not perform a 100% check of the AADs. As we have already explained, verification of each AAD is very labor intensive. Some companies are not checked for this reason. This violates the pattern, which requires 100% execution of the witness and verification activities. Therefore, we place *Y/N* in items VIa and V-II to indicate that these principles are not always observed.

Finally, the witnessing of export is not done simultaneously with the *Cross Border* activity, as prescribed by the pattern, but when the beer arrives at the premises of Retailer UK. As Figure 13.2 shows, the *Witness Export* activity occurs *after* the *Cross Border* and *Receive Beer* activities. The execution of the *Witness Export* activity after the *Cross Border* activity is only allowed if the outcome of the latter activity remains unchanged (see description of the Witnessing principle in section 2.1.1 for more details). This means that the contents of the container or beer truck should not be changed by unauthorized persons, e.g. by the carrier. This is not the case in this situation. In practice, the container can be unsealed and its contents can be changed. For example, some part of the cargo can be removed and remain undeclared at the customs.

Another well-known problem is that other goods e.g. drugs, can be smuggled in beer containers. In addition to the described control weakness, such a process creates a slowdown. Generally, customs waits about three month for AAD and if it has not been received after that time they can begin an investigation into fraud.

For all these reasons, the simultaneous execution of the *Cross Border* and *Witness Export* activities is required by the pattern. This is not the case, as the principle V-IIIa in Table 13.1 indicates.

Control models of the Tampered AAD scenario

In this step we redesign the ideal value and process models so that they comply with the Penalty pattern. The resulting models prescribe a control mechanism to combat the control problems

[4]The penalties exist in reality, but for illustrative purposes we do not model them explicitly in the ideal value model in Figure 13.2

of the Tampered AAD scenario exactly as suggested by the Penalty pattern. In addition, we require simultaneous execution of *Witness Export* and *Cross Border* activities.

Process view

Figure 13.5 shows the result of correcting the process model in Figure 13.2 so that it satisfies the Penalty pattern. Firstly, to remove verification slowdown, we require the *Witness Export* activity to be executed at the same time as the *Cross Border* activity. In Figure 13.2, we only model the parallel execution *Cross Border* and *Witness Export* with the transition UML element, since we cannot represent the simultaneous execution with UML.

As was explained earlier, Customs NL cannot witness exports, since the borders in the EU are open. Therefore, we should assign the *Witness Export* activity to an actor, who complies with the requirements of the pattern. For the moment, we introduce a hypothetical actor *trusted third party (TTP)*. This actor should be independent and socially detached from BeerCo NL, and, therefore, trusted by the Customs NL. In addition, this actor must also be trusted by BeerCo NL, e.g. because BeerCo NL shares sensitive information with this party. Therefore, we call it 'trusted *third* party', and not just 'trusted party'.

After witnessing export, the TTP sends *Export Evidence* to Customs NL. The *Export Evidence* is a *To-be-verified Statement* which contains information about the beer exported by BeerCo NL as witnessed by TTP. Finally, since the witnessing activity has been delegated to the TTP, the *Testify Export* activity is also added after the *Witness Export* activity, according to the Simple Delegation pattern.

Value view

The control value model is shown in Figure 13.6. We add the new actor TTP whose role is to perform the witnessing of the *Cross Border* activity and testifying to it to Customs NL. At the value level, we model this with a value object *Testify Export*, transferred from the TTP to Customs NL. In return, Customs NL provides *Service Value* to the TTP. The object *Service Value* can be a payment, if the service is provided by a commercial party.

In addition, in Figure 13.6 we model a value object *Export Evidence* and *Sub-ideal Export Evidence* instead of *AAD* and *Fake AAD*. This creates a more general model, in which evidence can be represented not only by AAD, but by any other statement that satisfies the pattern. Otherwise, the value model is the same as in Figure 13.1.

In the presented control models, the *TTP*, *Service Value*, and *Export Evidence* should be treated as *variables*. They can be substituted by real actors and documents. For example, in the AAD scenario, the role of *TTP* is played by Customs UK and the role of *Export Evidence* is performed by the *AAD*. Stamping the AAD corresponds to the *Testify Export* activity. In return, Customs NL provides Customs UK with similar data needed to verify exports from the UK to the Netherlands. This data provided in return represents the *Service Value* value object.

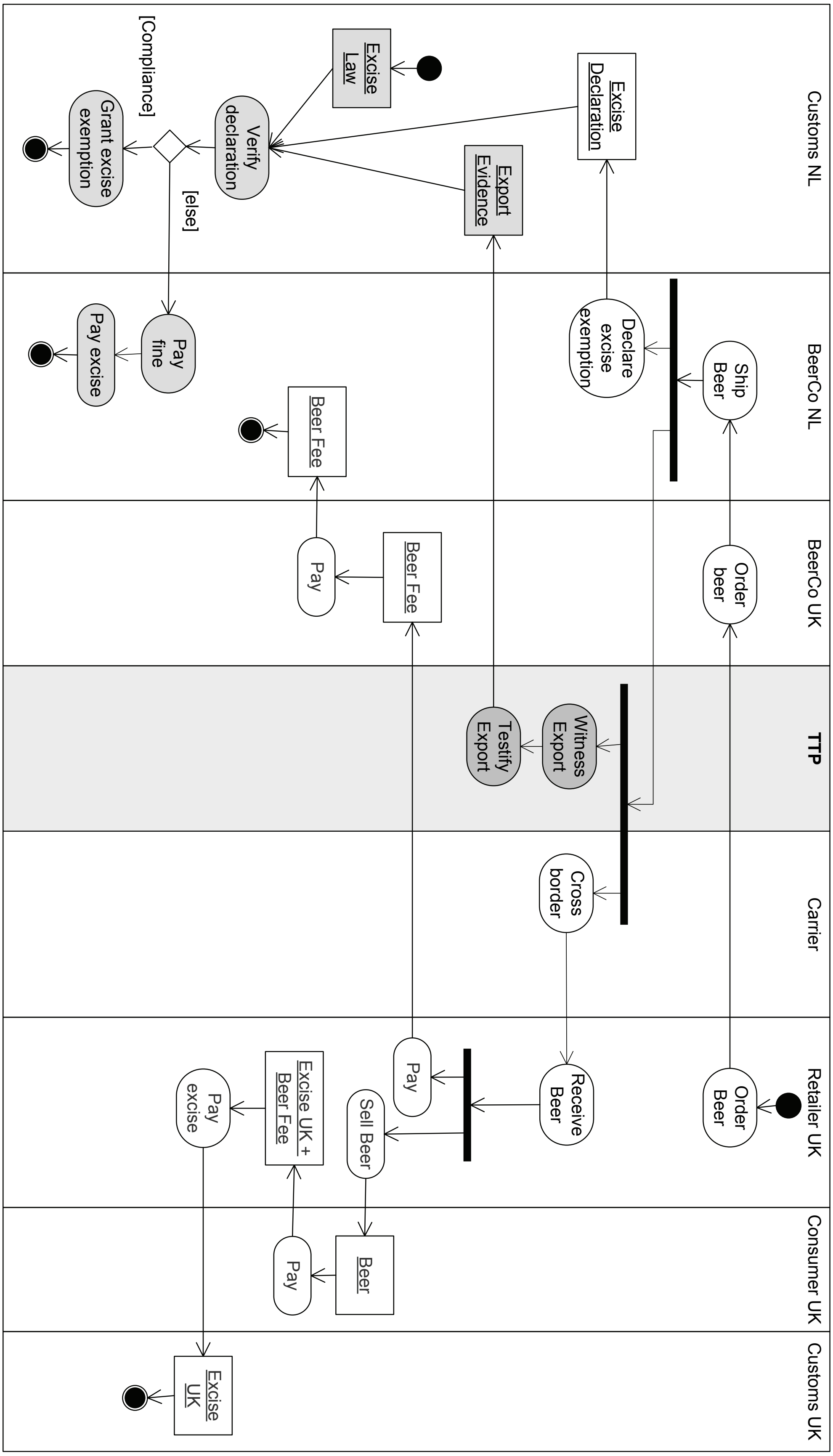


Figure 13.5: A control process model for the Tampered AAD scenario

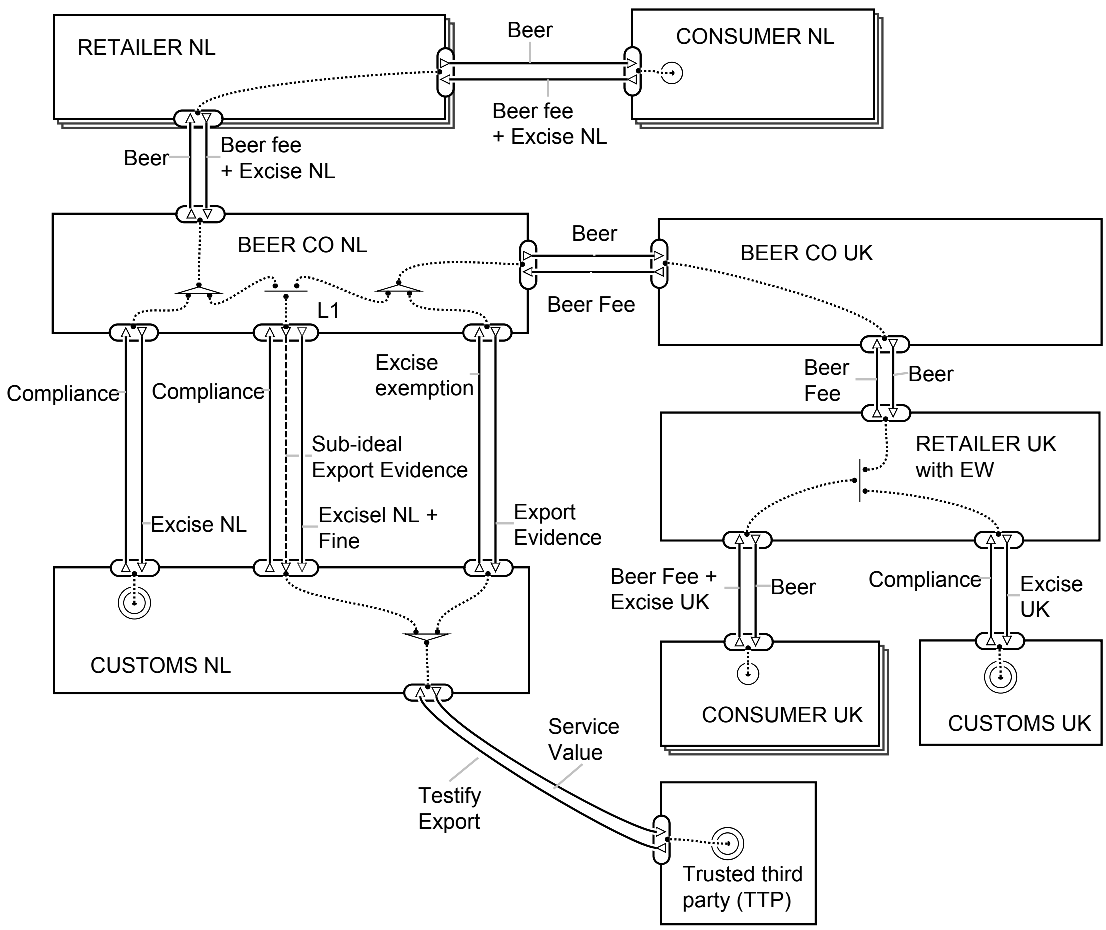


Figure 13.6: A control value model for the Tampered AAD scenario

13.5 EMCS scenario

In this section, we demonstrate another procedure that is currently being developed to replace the AAD. We derive this procedure by instantiating the generalized elements in the presented control models.

Currently, European governments and businesses are developing ICT-enabled solutions to cope with the described deficiencies in international trade procedures. One such solution is the Excise Movement Control System (EMCS). This is a system in which every EU customs authority records their data on excise goods entering the country. Using the EMCS system any EU customs department will be able to verify excise-free exports by consulting the EMCS system of the other EU customs departments. Thus, instead of using the AAD, Customs NL receives an electronic message from the EMCS system of Customs UK.

Process models

Figure 13.7 shows the models of export with the EMCS in place. Unlike in the process with the AAD the parties work with electronic messages. After goods are received by Retailer UK, he sends a message to Customs UK, thus testifying to receipt of the goods. After that Customs NL can consult the EMCS system, and receive a record from it with, e.g. all the imports of beer to the UK from the Netherlands during a certain period. So, the object *Export Evidence* from the control model in Figure 13.5 is instantiated with an *EMCS record*.

As we can see, the EMCS system improves the process. Firstly, it eliminates the indirect transfer of the export evidence, which is typical for AAD. The EMCS record is *directly* transferred from the trusted third party Customs UK to the primary actor Customs NL. Secondly, because the *EMCS Records* are transferred electronically, Customs NL are able to issue excise exemptions based on the electronic records. The verification of the electronic EMCS record is less labor intensive than the reconciliation of the paper AAD documents. As a result, all the principles from Table 13.1 are obeyed.

Value models

The value model of the EMCS scenario is shown in Figures 13.8. As in the process model, the role of the TTP is performed by Customs UK who deliver the EMCS service, which correspond to the value object Testify Export. Since the witnessing of the beer export is first done by Retailer UK, the *Testify Export* value object is transferred to Customs UK from Retailer UK. This value transfer corresponds to the process activity *Report Export* of Retailer UK in Figure 13.7. In return, Customs UK obliges the retailer to provide the information about the exports and to provide compliance with the obligation, see the value object (see *Compliance*). In this way, Customs UK *delegate* execution of the *Testify Export* service to Retailer UK.

Although the EMCS system complies with the Penalty pattern, it is still not perfect. Consider the following points:

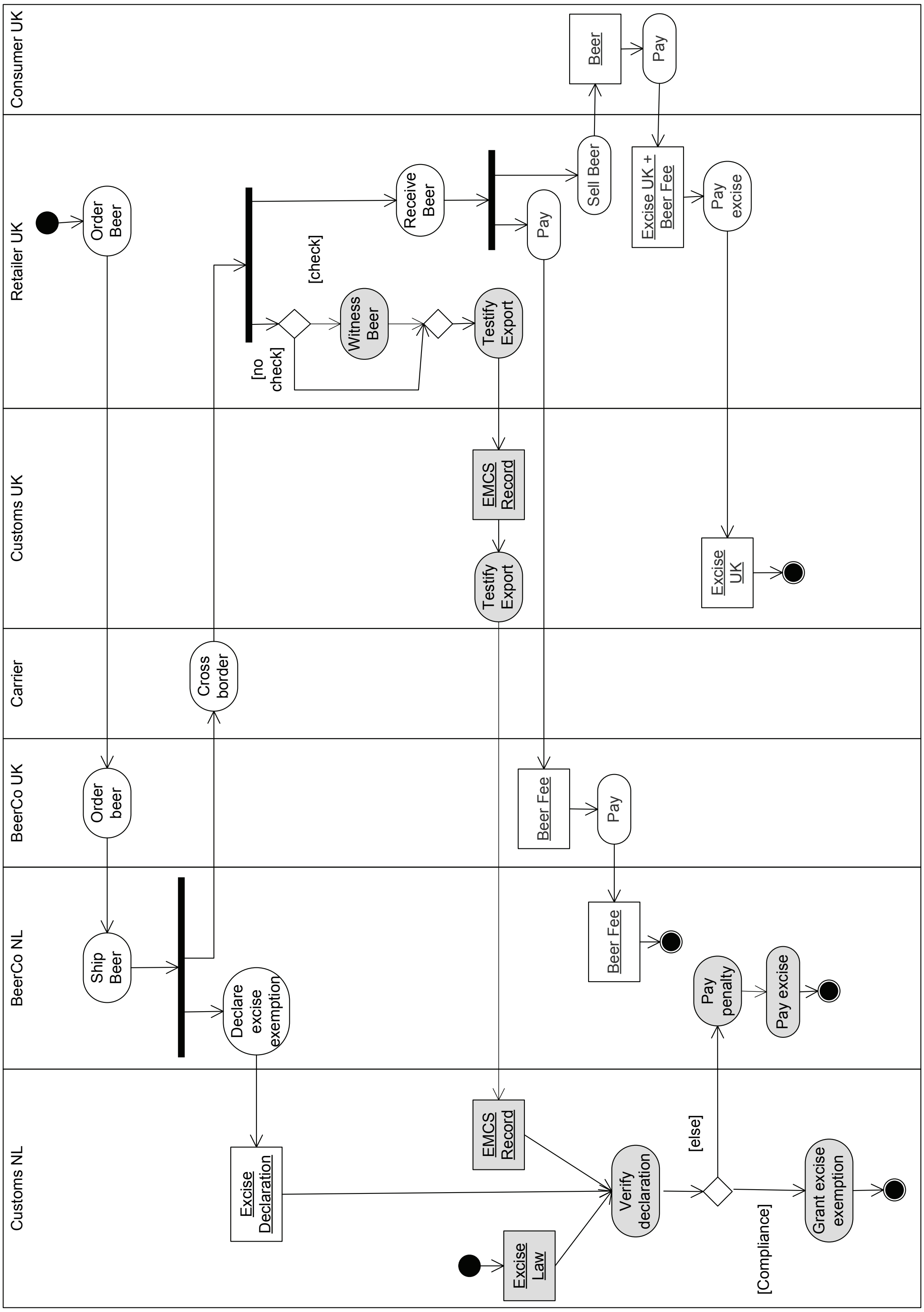


Figure 13.7: A process model with EMCS

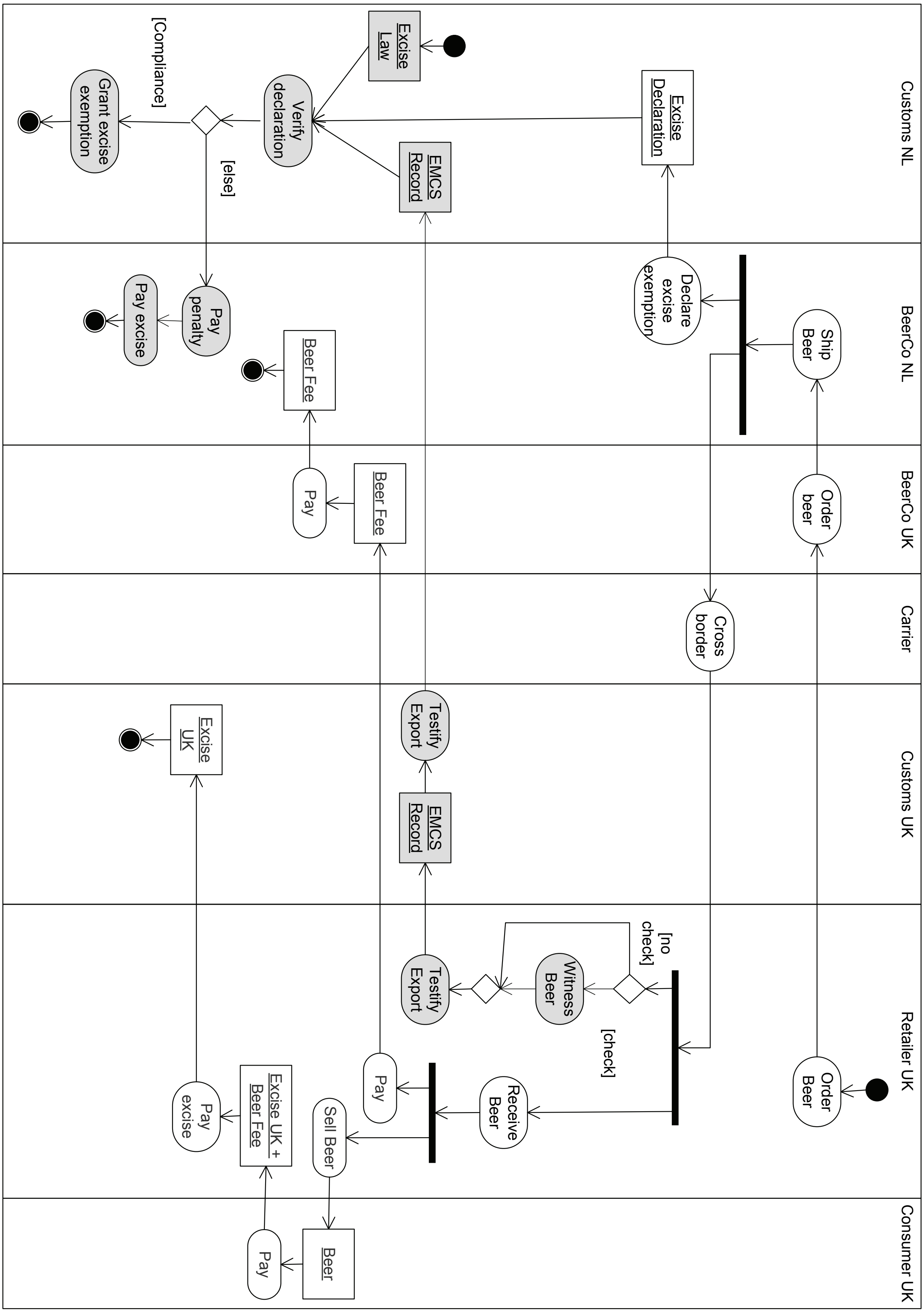


Figure 13.8: The value model with EMCS

- The activity *Witness Export* by Retailer UK is performed after the *Cross Border* activity. This is not reliable since the contents of the container can be changed before it arrives at Retailer UK. In addition, it creates a slowdown in the process.

- The AAD cannot be eliminated even if the EMCS system is in place. The AAD is still required for controls *en route*.

- The control relies on the EW licensing of retailers.

13.6 Virtual Beer scenario

In this section we demonstrate why EW licensing is needed and whether the patterns are able to suggest the *Excise Warehouse* (EW) licensing mechanism. The role of this licensing will be explained in the following section by means of the Partner Screening pattern. Therefore, we further assume that Retailer UK has no EW license[5].

One of the risks that occur, if the EW licensing is not in place, is the so-called 'virtual beer' fraud. The term 'virtual' refers to the fact that beer export does not take place physically, but only 'on paper'. In our case, we consider that the beer is 'virtually' shipped to Poland, when actually it is sold in the Netherlands. The difference with the Tampered AAD scenario considered in the first e^3*control* cycle is that BeerCo NL is not the fraudster. The fraudsters are the buyers in Poland and a retailer in the Netherlands.

The goal of the scheme is to avoid excise payments in a country with high excises, and pay them in the country with low excises. In our example, the excises are paid in Poland, instead of the Netherlands. In some cases, excise payments in both countries can be avoided, e.g. by bribing the customs officers in Poland.

In some cases, a Retailer PL is formed by BeerCo PL and Retailer NL only for the sake of avoiding the excise payments. Retailer PL is created only for one-two transactions and disappears (goes bankrupt) as soon as the money has been laundered. Obviously, such a retailer cannot be trusted to testify to exports, which demonstrates the weakness of the EMCS and AAD procedures should the EW licensing of retailers not be required.

The 'virtual beer' scenario is possible in a situation in which BeerCo PL pays BeerCo NL at *ex-works delivery permissions*, meaning that as soon as the goods leave the premises of BeerCo NL on Dutch territory, possession is already in the Polish company. Unlike in the previous scenario, the carrier now works on behalf of BeerCo PL.

There are multiple controls that can help to detect or prevent 'virtual' beer exports. We analyze the problem and discover such controls, by using the patterns. As already stated, we deliberately assume that the Retailer in Poland has no status of Excise Warehouse.

[5]According to the legislation, if the retailer in the country of import has no EW license, the whole excise-free procedure is prohibited. However, we make this assumption here for illustrative reasons to be able to demonstrate why EW licensing is needed and how the patterns describe the EW licensing mechanism

13.6.1 Step 1: Design ideal situation

As a starting point in our analysis we assume that the controls of the EMCS procedure are in place. As in the EMCS procedure, the witnessing of the export is delegated to Customs NL, who in its turn delegates it to Retailer UK. The ideal value model is shown in Figure 13.8. The ideal process model is shown in Figure 13.7.

13.6.2 Step 2: Design sub-ideal situation

Value view

As in the ideal value model, the sub-ideal value model, shown in Figure 13.9 has two consumer needs: one for sales in Poland and one for sales in the NL. The Virtual Beer scenario considers sales of the beer in the Netherlands. Therefore, the sub-ideal path starts at Customer NL.

Retailer NL has two choices of selling the beer in the Netherlands, as indicated by the *OR* fork. Firstly, he can sell it legally by paying the excises in the Netherlands. This corresponds to the left path of the *OR* fork. In this case, Retailer NL buys beer from BeerCo NL.

Secondly, Retailer NL can buy beer illegally and sell it illegally. In such a way, Retailer NL gets excise payments from the customers but does not transfer them to BeerCo NL. As a result, Customs NL does not receive excise payments. This choice of Retailer NL corresponds to the right path of the *OR* fork. In this case, Retailer NL buys beer from BeerCo PL. This transaction corresponds to value transfers *Smuggled Beer* and *Smuggled Beer Fee* between BeerCo PL and Retailer NL. The beer is shipped physically from BeerCo NL to the Carrier, who acts on behalf of the BeerCo PL. BeerCo PL buys beer from BeerCo NL, as in the ideal scenario.

The transaction with the *Smuggled Beer* takes place physically, but it is no reported in the administration of BeerCo PL or Retailer NL. In fact, BeerCo PL registers a transaction with a retailer in Poland Retailer PL. To model this, we introduce the value objects *Virtual Beer* and *Virtual Beer Fee* between BeerCo PL and Retailer PL. We call these objects ‘virtual’ since they do not correspond to any physical shipments and sales of the beer in Poland.

The value transfers *Virtual Beer*, *Virtual Beer Fee*, *Smuggled Beer* and *Smuggled Beer Fee* are sub-ideal and are marked with dashed lines. The parties Retailer PL, Retailer NL and BeerCo PL are also given liability tokens.

BeerCo NL does not know that the beer they sell to the Polish company is sold in the Netherlands. Obviously, BeerCo NL expects to have the *Export Evidence* (e.g. EMCS record or AAD) stamped by Polish customs. To ensure this, Retailer PL testifies to Customs PL about the delivery of the ‘virtual’ beer from the Netherlands. As a result, Customs PL issues the *Export Evidence* for ‘virtual’ beer, called *Virtual Export Evidence*. As in the ideal model, BeerCo NL delivers the *Virtual Export Evidence* to Customs NL and receives *Excise Exemption* in return.

Although the value object *Virtual Export Evidence*, exchanged by BeerCo NL, is sub-ideal, BeerCo NL does not get the liability token. This is because BeerCo NL does not behave sub-ideally. The liability tokens are only assigned to the colluding parties BeerCo PL, Retailer NL and Retailer PL, since they are responsible for the whole scheme.

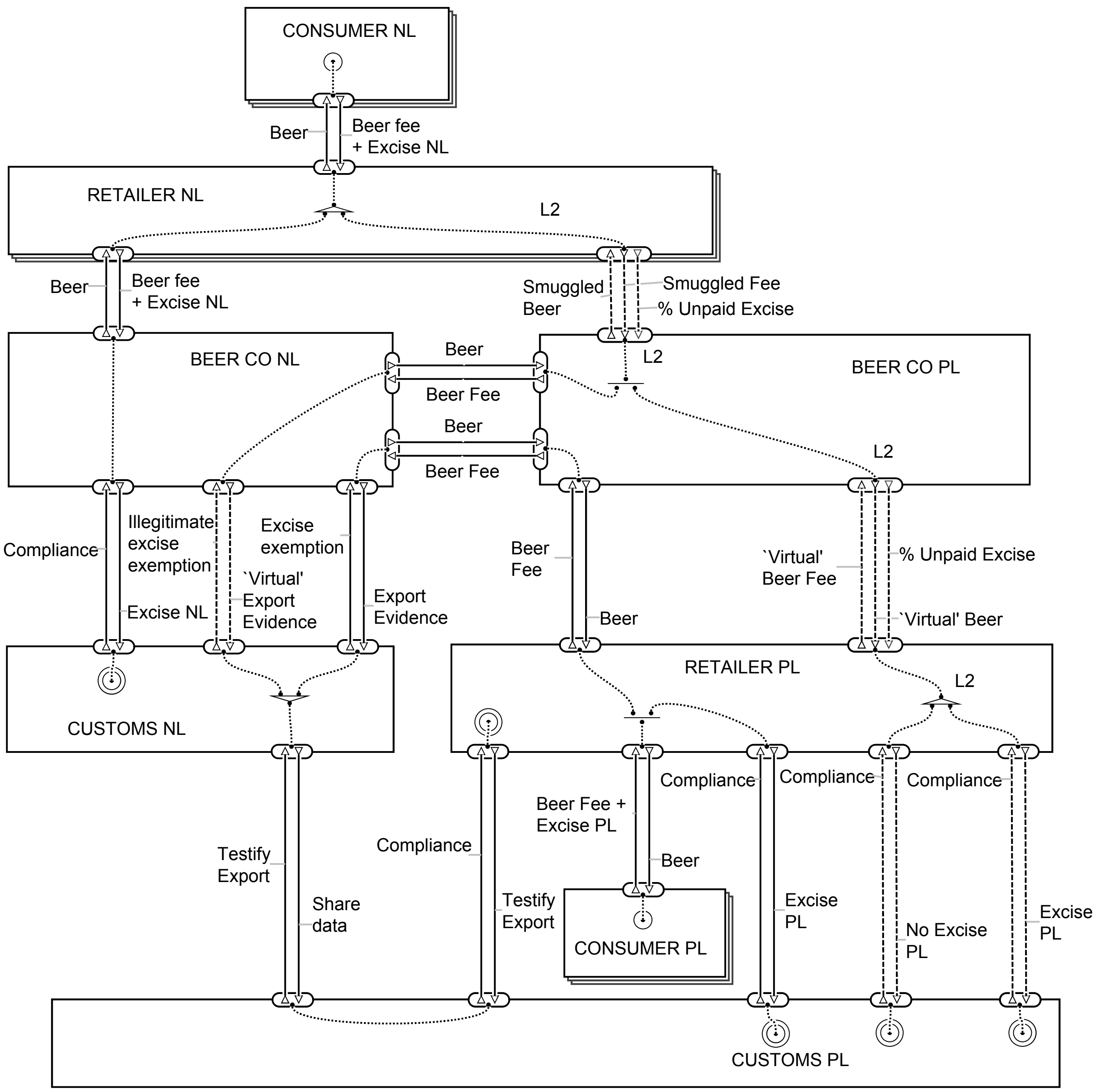


Figure 13.9: Virtual Beer scenario: a sub-ideal value model

The gain for the colluding parties in the sub-ideal scenario is the avoided excises. The excises in the Netherlands are always avoided, since these transactions are not recorded and are unknown to the authorities. Polish excises for the ‘virtual’ beer are either paid or not. This choice is modelled with the *OR* fork at Retailer PL. If excise in Poland is paid, Retailer PL and Customs PL exchange objects *Excise PL* and *Compliance*. Otherwise, they exchange *No Excise PL* and *Compliance*. These value objects are also sub-ideal (even if the excise is paid), since they do not correspond to actual sales within Poland.

The gain is shared by BeerCo PL, Retailer NL and Retailer PL and is modelled in Figure 13.9 with a value object *%Unpaid Excise*. The % sign represents that each actor receives some part of the unpaid excises. If the parties choose to pay excises in Poland, the profit is the difference between the excises in the Netherlands and Poland: $\%Unpaid\ Excise = \%(ExciseNL - ExcisePL)$. This scenario is only profitable if the excise tariff in Poland is lower than the tariff in the Netherlands. If the parties choose to avoid paying excises in Poland, the revenue of the parties is the unpaid excise in the Netherlands: $\%Unpaid\ Excise = \%ExciseNL$.

Process view

Figure 13.10 represents a sub-ideal process model of the Virtual Beer scenario. For simplicity of visualization we assume that payment of excises in Poland is avoided.

In this process, the beer is ordered from BeerCo NL by Retailer PL via BeerCo PL. Physically the beer is delivered by the Carrier PL to Retailer NL who then sells the beer and receives money from the Dutch consumer. Then Retailer NL shares the profits with the BeerCo PL and Retailer PL, from which BeerCo PL repays to Beer Co NL. Retailer PL also testifies about the export to the Customs PL, as a result of which, BeerCo NL receives the export evidence and gets the excise exemption.

The process differs from the ideal process model in Figure 13.7 in several aspects. Firstly, the carrier does not cross the border (see activity *Do Not Cross Border*). Secondly, the physical beer is not received by Retailer PL, but by Retailer NL (see activity *Receive Beer* at Retailer NL). Consequently, Retailer NL sells the beer to Consumers NL, not to Consumers PL (see activity *Sell Beer* at Retailer NL). Retailer PL does not make a payment to BeerCo PL, as in the ideal model. Instead, it fakes a record of payment in its administration (see activity *Record Virtual Transaction* at Retailer PL). Furthermore, the payments are repaid between the colluding parties. Retailer NL, who executes the ‘black’ sales and thus receives all the money from the consumers, repays *Beer Fee* and shares the ‘black’ profit *%Unpaid Excise* with BeerCo PL (see activities *Pay Beer* and *Share Illegal Profit* at Retailer NL). BeerCo PL also shares the illegal profit with Retailer PL (see activity *Share Illegal Profit* at BeerCo PL). All these activities are sub-ideal.

13.6.3 Step 3: Select a control pattern

To solve the control problem, we add a control mechanism. In this step we select patterns that are appropriate to address the control problem of the ‘virtual beer’ scenario.

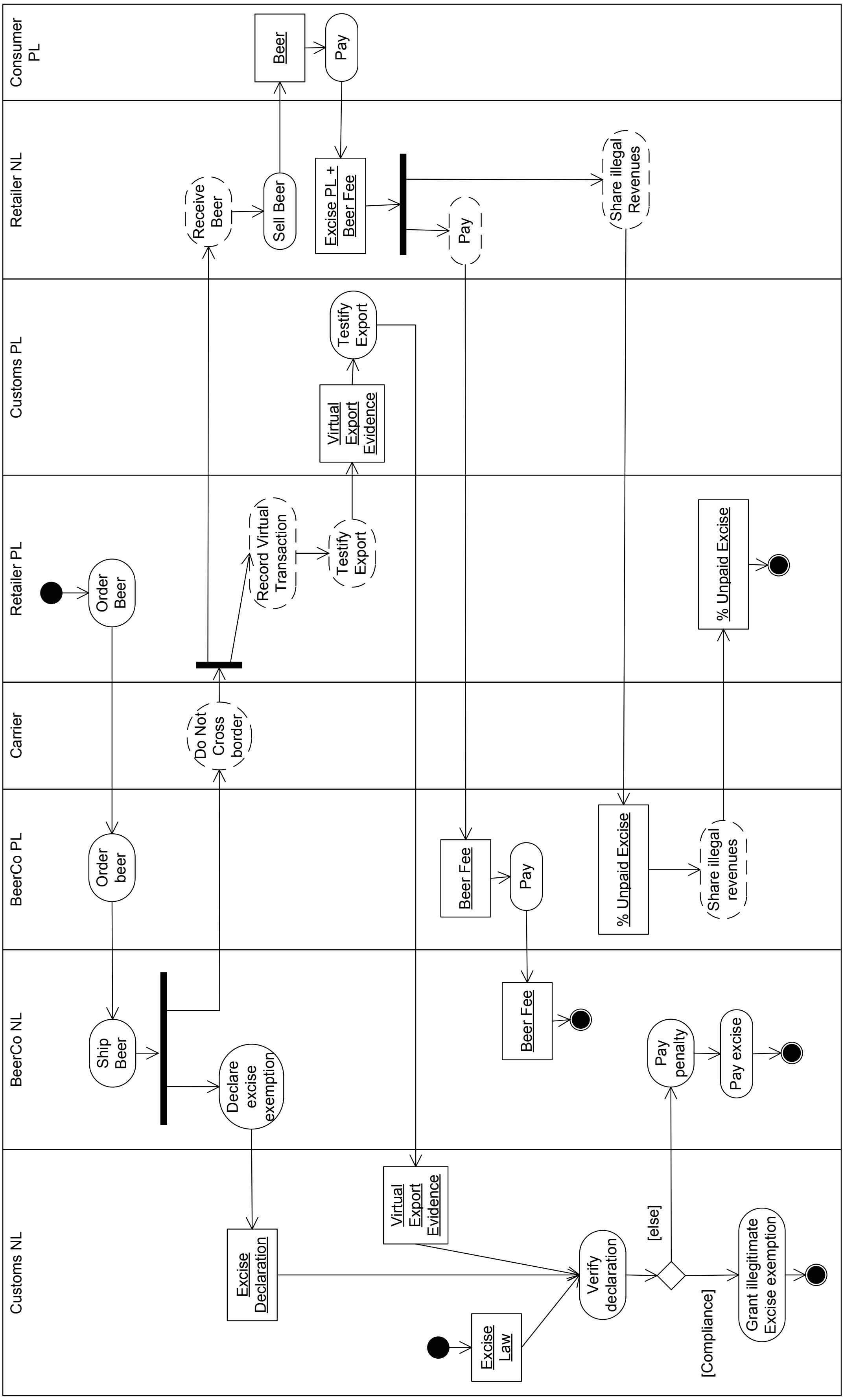


Figure 13.10: Virtual Beer scenario: a sub-ideal process model

Identify primary actor, counter actor, primary object and counter object, ideal and sub-ideal primary and counter activities

To find a pattern for the problem at hand, we first need to identify a primary actor, counter actor, primary value object and counter value object. The next step is to identify primary and counter activities. For ideal activities, we use the ideal process model. For sub-ideal activities, we use the sub-ideal process model.

Primary Actor. As in the Tampered AAD scenario, Customs NL is the primary actor.

Counter Actor. The Virtual Beer scenario has several counter actors: Retailer NL, Retailer PL and BeerCo PL. They all behave sub-ideally and have the liability token **L2**.

Sub-ideal Counter Value Object. There are numerous counter value objects, including *Virtual Export Evidence*, *Virtual Beer*, *Virtual Beer Fee*, *Smuggled Beer*, *Smuggled Beer Fee*, *%Unpaid Excise*, and *No Excises PL* . All the sub-ideal counter value objects are marked with the dashed lines in Figure 13.9. They are also listed in Table 13.2.

Ideal Counter Value Object. The ideal counter value object for the sub-ideal *Virtual Export Evidence* is the object *Export Evidence*, the ideal value objects, which correspond to the sub-ideal objects *Virtual Beer*, *Virtual Beer Fee* and others do not exist, since the transaction does not happen in the ideal situation.

Sub-ideal Primary Value Object. The sub-ideal primary value object is the value object *Excise Exemption*, exchanged in return for the sub-ideal counter value object *Virtual Export Evidence*.

In this case, there are multiple counter value objects, including *Virtual Export Evidence*, *Virtual Beer*, *Virtual Beer Fee*, *%Unpaid Excise*, etc. However, since only one counter value object *Virtual Export Evidence* is exchanged in a *direct* value transfer with the primary actor Customs NL, we only consider this object in order to find the corresponding sub-ideal primary object. This is an unusual situation, which occurs because none of the counter actors has a direct value transaction with the primary actor Customs NL.

Ideal Primary Value Object. As in the Tampered AAD scenario, the ideal primary value object is *Excise Exemption*, exchanged in return for the ideal counter value object *Export Evidence*.

Primary Activities. As in the previous scenario, the *Grant Excise Exemption* is the primary activity.

Sub-ideal Primary Activities. If *Virtual Export Evidence* is presented, the activities corresponding to the granting of the excise exemption are sub-ideal. So, the activity *Grant Illegitimate Excise Exemption* in Figure 13.4 is sub-ideal.

Sub-ideal Counter Activities. Because there are multiple sub-ideal value objects, there are also multiple ideal and sub-ideal counter activities. The sub-ideal counter activities are marked with the dashed lines in the sub-ideal process model in Figure 13.10. Table 13.2

Table 13.2: Sub-ideal counter value objects and related sub-ideal counter activities in the 'Virtual Beer' scenario

Sub-ideal Counter Value object	**Coming from**	**Activity**	**Executed by**
Virtual Export Evidence	BeerCo NL	*Do Not Cross Border*	Carrier
		Testify Export	Retailer PL
Smuggled Beer Fee	Retailer NL	*Pay*	Retailer NL
Smuggled Beer	Retailer PL	*Receive Beer*	Retailer NL
Virtual Beer Fee	Retailer PL	*Record Virtual Transaction*	Retailer PL
Virtual Beer	BeerCo PL	*Record Virtual Transaction*	Retailer PL
		Do Not Cross Border	Carrier
%Unpaid Excise	Retailer NL	*Share Illegal Revenues*	Retailer NL
%Unpaid Excise	BeerCo PL	*Share Illegal Revenues*	BeerCo PL
No Excise PL	Retailer PL	–	–

shows more precisely which sub-ideal counter activities are related to the transfers of sub-ideal counter value objects. The sub-ideal counter activities include *Do Not Cross Border*, *Share Illegal Revenues*, *Record Virtual Transaction*, etc.

Identify type of the sub-ideal activity and patterns

To select a pattern, we must first identify the *type* of sub-ideal activities. In Chapter 7, we have distinguished three types of sub-ideal activities: Sub-ideal Counter, Emerging Sub-ideal, Deny Commitment and Deny Execution (see Chapter 11 for more details).

There are multiple sub-ideal activities. They are all of type Sub-ideal Counter or Emerging Sub-ideal activity. For example, the sub-ideal *Do Not Cross Border* is the not executed ideal *Cross Border* activity; so, this activity is of type Sub-ideal Counter. The sub-ideal *Receive Beer* is the ideal *Receive Beer* activity executed by a wrong actor; thus, this activity is of type Sub-ideal Counter. *Record Virtual Transaction* corresponds to the recording of a real transaction; this activity does not correspond to any activity in the ideal value model and is thus an Emerging Sub-ideal activity. *Share Illegal Revenues* is also an Emerging Sub-ideal activity.

The types Sub-ideal Counter and Emerging Sub-ideal signal the presence of the control problems described in the Partner Screening, Execution Monitoring, Penalty, and Incentive patterns (see Chapter 11 for more details). Before selecting a pattern, we will first discuss the application of each possible option.

13.6.4 Step 4: Applying patterns

Penalty pattern

The Penalty pattern suggests decreasing the value the counter actors BeerCo PL, Retailer PL and Retailer NL get from the sub-ideal path. To do this an additional outgoing value object *Fine* could be introduced to decrease the value accumulated by them from getting the *%Unpaid Excise*. However, ex-post fines are only effective if the fraud can be easily detected. This is not the case with the virtual beer scenario. As we have explained earlier, when the authorities discover the fraud and decide to collect the excise, the fraudulent parties have already left. Another possibility is presented by the Incentive pattern, which suggests adding an incoming value object to the counter actors, e.g. by rewarding them for paying correct excises.

Another opportunity, which can be modelled as the Penalty and Incentive patterns, is to make the excise tariff in Poland and the Netherlands the same. If the tariffs are the same, there is no point to such a fraud unless the excise in the country of 'virtual' export (here: Poland) is avoided completely. It is current EU policy to make excise tariffs equal through-out the EU. However, this solution does not eliminate the financial gain from fraud if excises are avoided in both countries.

Execution Monitoring pattern

An Execution Monitoring pattern suggests detecting at least one of the multiple sub-ideal counter activities. For example, detection of the activity *Record Virtual Transaction* of Retailer PL means that Customs NL must check whether all transactions at Retailer PL have corresponding transactions with their customers. In this way they can discover records of the virtual transactions. However, this procedure is time consuming and even unrealistic since the retailer might have many customers.

Detection of the activity *Receive Beer* of Retailer NL considers catching the Retailer NL moving beer from a truck to his premises. However, in this case the parties can present a forged AAD, which states that Retailer NL bought the beer.

Customs may try to catch the parties exchanging the 'black' money through controlling activities *Share Illegal Revenues*. For example, if they identify illegal money, they can the parties owing the money.

The most efficient way is to apply the Execution Monitoring pattern to detect the *Cross Border* activity, as in the Tampered AAD scenario. However, it remains difficult to find a party that could provide trustworthy export evidence; unless the EW licensing is implemented.

Partner Screening pattern

The *Excise Warehouse* licensing of retailers can be described by means of the Partner Screening pattern which suggests 'screening' the sub-ideal actors (Retailer NL, BeerCo PL and Retailer PL) before allowing them to operate with the excise-free procedure. Because the involvement of Retailer NL in the virtual beer procedure is not recorded anywhere, it is only feasible to screen BeerCo PL or Retailer PL, with whom BeerCo NL has official transactions. The screening of Retailer PL results in the EW licensing process. This model is shown in Figure 13.11.

The model shows that before issuing an *EW license*, Customs PL audits the internal processes of Retailer PL. In accordance with the Partner Screening pattern, we have added a *Witness* activity to Customs PL. The witnessing refers to the observation of certain *Internal Controls* of the retailer. The witnessing produces the to-be-verified statement *Internal Control Description*, which describes the processes as observed by Custom PL. After that, Customs PL performs the verification activity *Audit Internal Controls*, by which the internal processes are compared with the supporting statement of required quality standards for excise warehouses, modelled with the object *EW Controls Standards*. If verification is positive and reveals that the internal processes comply with the *EW Controls Standards*, Customs PL grants Retailer PL the status of Excise Warehouse by executing the *Certify EW* activity and providing the *EW License*.

The audit is carried out by Customs PL, not Customs NL, which means that Customs NL delegates the verification activity *Audit Internal Controls* to their colleagues in Poland. Although Customs NL are primarily interested in controlling Retailer PL, Customs NL have no legal rights to oblige Retailer PL to comply with all rules. According to the pattern, Customs NL should only execute the primary activity *Grant Excise Exemption* if the *Certify EW* activity is executed.

In fact, excise-free export should not even start if Retailer PL is not EW-licensed. To model this, we show that the whole process of excise-free export as one activity called *Excise-Free Export*. In this model EW licensing is performed one time, while the export activities, including the primary activity *Grant Excise Exemption*, are executed multiple times. This situation is specific for this model and makes it different from the original Partner Screening pattern.

The corresponding value model of EW licensing is shown in Figure 13.12. At the value level, the transfers *EW License* and *EW Standards Compliance* between Customs PL and Retailer PL have been added. The object *EW Standards Compliance* refers to the compliance of internal controls of Retailer PL with EW control standards. *EW License* shows that Retailer PL has the Excise Warehouse license.

13.7 Future scenario with TREC

A combination of the AAD procedure with EW licensing still contains control weaknesses, since the AAD is vulnerable to tampering. One option to improve the state of affairs is to replace paper-based procedures by electronic ones. This is what the EU is currently doing by introducing EMCS instead of the paper-based AAD. The combination of the EMCS systems with the EW licensing procedures is not yet perfect. Firstly, the witnessing activity *Witness Export* is performed by Retailer UK *after* the *Cross Border* activity. This is not reliable since the contents of the container can be changed before it arrives at Retailer UK. Secondly, the AAD cannot be eliminated and is still needed for controls *en route*.

Much greater benefits can be achieved if a radical rethinking takes place and the assumptions underlying the procedures are questioned. The ITAIDE project has opted for this approach. Two

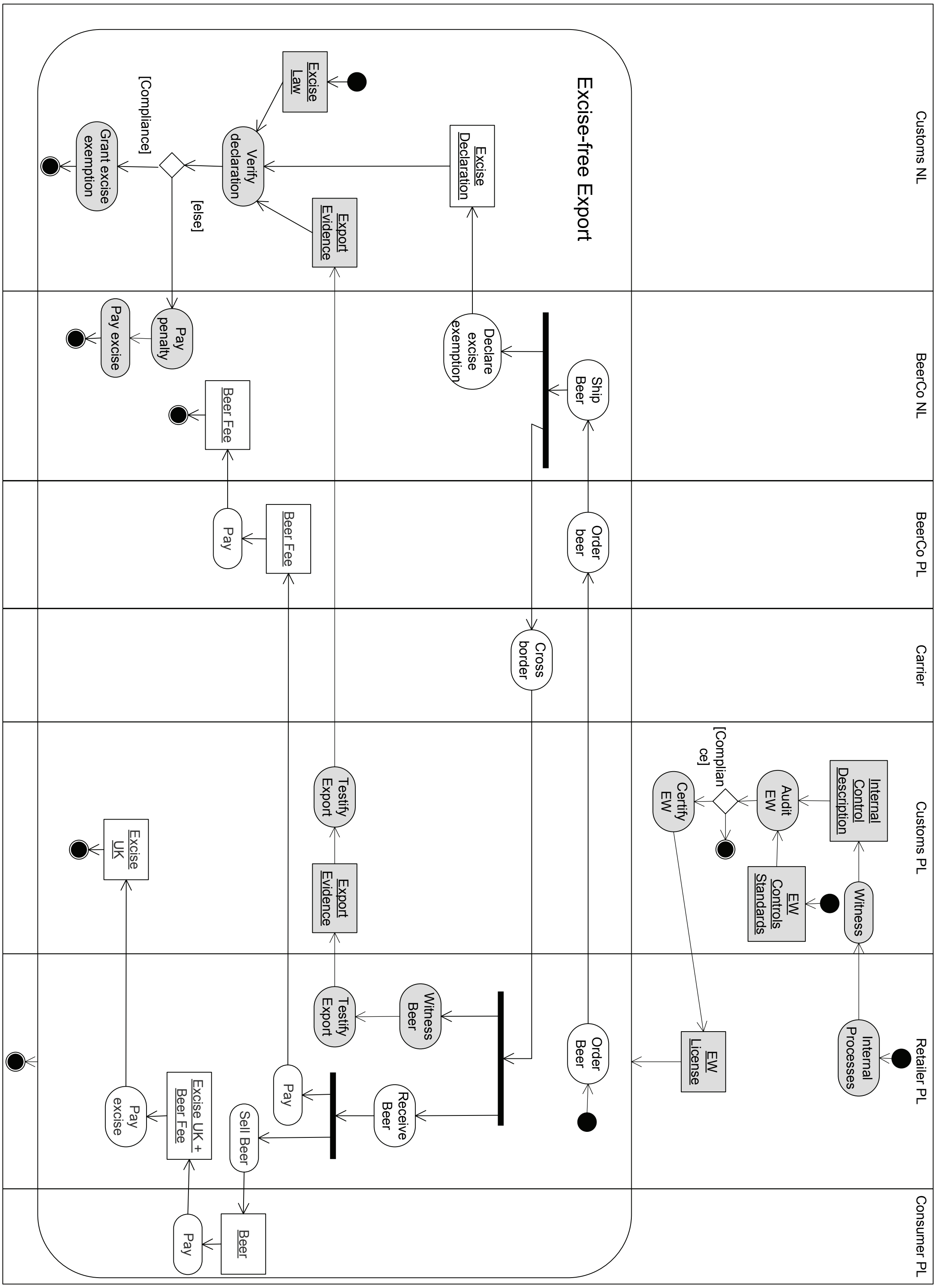


Figure 13.11: A process model with licensing of the Excise Warehouse

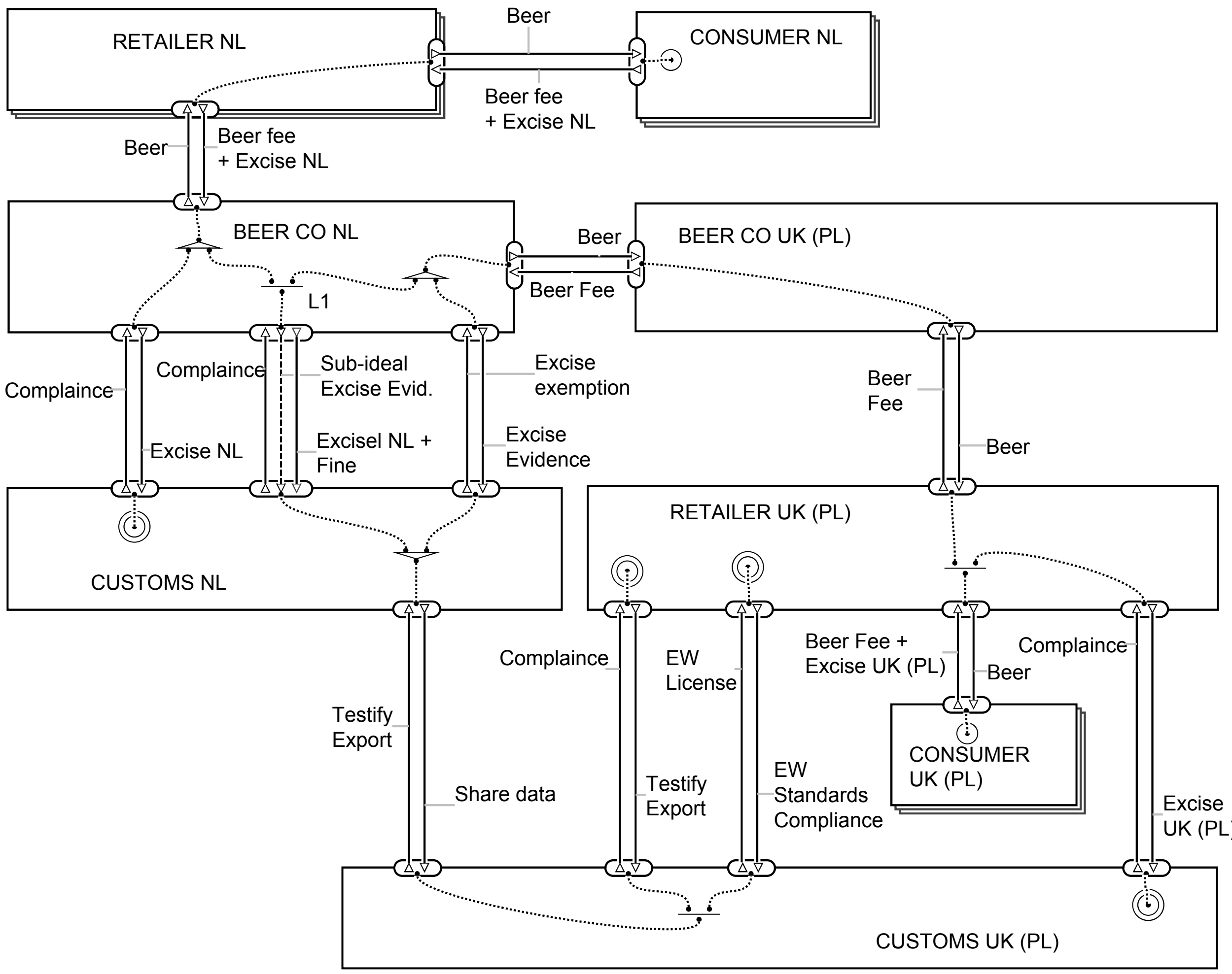


Figure 13.12: A value model with licensing of the Excise Warehouse

technologies are used as corner stones in the BeerLL export procedure: the TREC smart seal for container security and Electronic Product Code Information Services (EPCIS) databases. The problems with the EMCS and the AAD can be solved by means of the BeerLL export procedure. We will now describe how the BeerLL export procedure works based on the description in [Baida et al., 2007]. After that we analyze this innovative trade procedure using the patterns.

13.7.1 Innovative Trade Procedure

The Tamper-Resistant Embedded Controller (TREC)[6] is a container-mounted device which has a mobile receiver tracking the containers precise location; sensors monitoring environmental parameters in the container (e.g., temperature, humidity), sensors monitoring the physical state of the container (e.g., door opening, tampering attempts) and communication modules for exchanging data (e.g., via handheld devices, via satellite, GSM/GPRS or short range wireless). By monitoring a container's position coordinates, an automatic message can be triggered by a TREC device to supply chain partners including Customs NL, when the container actually leaves the Netherlands, deviates from its predefined route, is being opened by an unauthorized party, or when other predefined events occur. By monitoring a containers location, TREC devices could replace the AAD and EMCS functionality to provide export evidence.

Container Information Services are leveraging the EPCglobal network and EPCIS non-proprietary standards currently under definition by EPCglobal[7]. Those standards define interfaces, discovery services, security mechanisms and other infrastructure for capturing and querying supply chain data (and other EPC related data). The EPCglobal network, also called the 'Internet of things, is a suitable backbone for tracking goods moving along a supply chain. It leverages the infrastructure from the Internet to create an open standard, Service Oriented Architecture-based data sharing mechanism between trading partners.

When BeerCo NL prepares a shipment of beer in the new procedure, it publishes the goods commercial data (originating from its ERP system) in its own EPCIS database that is accessible through the Internet for authorized supply chain partners, including Customs NL. As soon as a beer container is closed at the premises of BeerCo NL, the TREC device on that container sends a message to the carrier and notification is sent to Customs NL. This message contains a Unique Consignment Reference number (UCR), which the carrier and customs can use to retrieve commercial data from BeerCo's EPCIS and use it for all their procedures, including excise, VAT, statistics and more. Hence, data is kept at BeerCo's EPCIS and is accessible for all relevant supply chain partners and government systems, also for periodic audits. As soon as a container physically leaves Dutch territory (or: arrives at the country of destination), the TREC device sends a message to Customs NL, providing digital export evidence. If the shipment is physically inspected en route, customs officers can use handheld devices to obtain access via the Internet and by using a UCR that the TREC device provides to commercial information which identifies this shipment in BeerCo's EPCIS. The digital export evidence (produced by a TREC device) is also stored in the carrier's EPCIS database, and can be accessed by authorized supply chain partners.

[6] Further information on TREC is available at $http : //www.zurich.ibm.com/news/05/trec.html$ and $http : //www.research.ibm.com/jam/secure_{t}rade_{l}ane.pdf$, last accessed on April 27, 2007.

[7] For further details see http://www.epcglobalinc.org

13.7.2 Analyzing the TREC scenario with patterns

In Figure 13.13 we adjust the control model developed in Figure 13.5 so that it fits the new solution. Note that we do not model the complete solution, but only a part of it. For example, we do not model the use of AAD for checks en route.

The TREC service provider is a party which operates the TREC device. In the terms of our model in Figure 13.5, the TREC service provider performs the role of the TTP. Furthermore, as soon as a container physically leaves Dutch territory or arrives at the country of destination, the TREC device sends a message to Customs NL, providing digital export evidence. In our terms, this means that TREC can execute the *Witness Export* activity simultaneously with the *Cross Border* activity, as prescribed by the patterns. In addition, according to the pattern, TREC service provider sends the *TREC Message* directly to Customs NL. The *TREC Message* plays the role of the *Export Evidence*.

This model has no control weaknesses and is implemented exactly according to the Penalty pattern. With this model we show that TREC devices could replace the functionalities of AADs or even EMCSs to provide more reliable evidence of export. In addition, TREC can also be used for controls *en route*.

A pre-requisite for this scenario, according to the patterns, is that the TREC service provider is independent and socially detached from BeerCo NL both at the company as well as at the employee level. This is to avoid the risk that an employee can send a message about crossing a border when no such event has taken place. A TREC device must therefore be able to send a message without human intervention. To ensure this, a licensing process for TREC service providers, similar to EW licensing is required. Licensing must ensure that only providers with reliable processes can operate as TREC service providers. Such licensing can be provided by customs or even commercial parties operating on behalf of customs.

The licensing of TREC services is modelled in Figure 13.13. It shows that Customs NL verifies compliance of the *TREC Processes* with a particular TREC service provider with established *TREC control standards*. If verification shows that the TREC service provider complies with control standards, he will receive the *TREC Licence* and may operate with excise-free exports.

At the value level (see Figure 13.14), the model is an instantiation of the model in Figure 13.6 with additional exchanges for the TREC service licensing. We show that a TREC service provider sends a message to Customs NL, while payment for the TREC service is done by BeerCo NL (see *TREC Service Fee*). Which actor will pay for the TREC service is still under discussion, but most probably it will be done by the commercial parties.

13.8 Discussion

In this chapter we have demonstrated how to put our patterns into practice in a case study about the redesign of customs procedures for excise duties. Our focus was on the way the parties that buy or sell excise goods can avoid excise payments in illegitimate way. As an example, we used the export of beer from the Netherlands to the UK and to Poland.

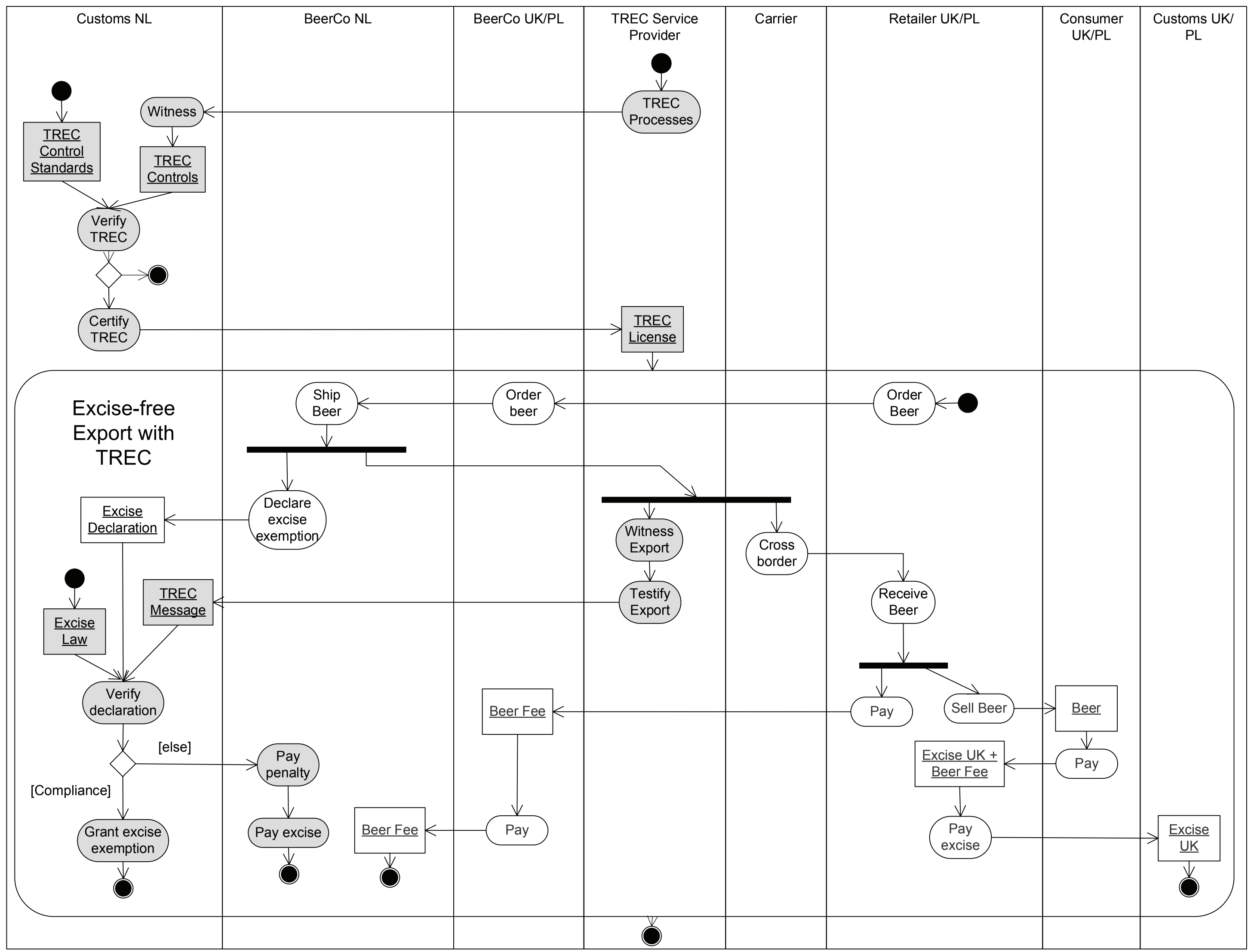


Figure 13.13: A process model with the TREC device

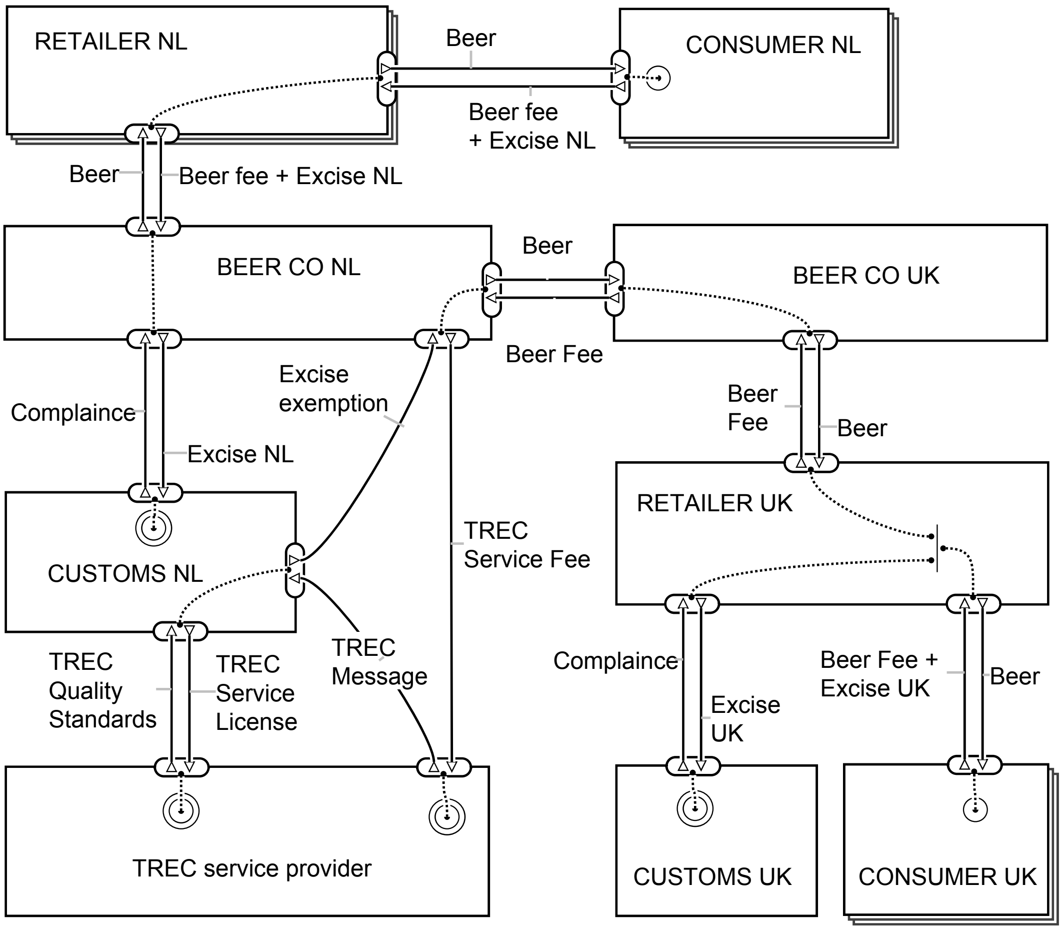


Figure 13.14: A value model with the TREC device

Based on interviews with domain experts, such as customs officers and managers of a beer producing company, we have discovered two common illegitimate ways of avoiding excise payments in the Netherlands. The first is when the beer producer in the Netherlands BeerCo NL overstates exported goods and understates goods sold in the Netherlands. This was referred to as the Tampered AAD scenario, since it involves tampering with the evidence document AAD. The second way to avoid excise payments is referred to by the domain experts as ‘virtual beer’. In this case, excises are avoided in the Netherlands; however the beer producer in the Netherlands is not involved in the fraud at all. Other parties, namely the buyers in the country of the importer (we use Poland as an example) and the local retailer in the Netherlands, collude to create exports on paper but in reality they sell the beer in the Netherlands on the ‘black’ market. In this way, the parties avoid excise payments either in the Netherlands, or even in both countries.

The case study revealed that the current EU AAD-based procedure is vulnerable to fraud with excise declarations. The AAD, which should prove that goods have indeed left the country, is transferred along all the parties of the supply chain, which makes it vulnerable to tampering. This control problem can be mitigated in several ways. One of these would involve a technical device, the TREC, and a new actor, the TREC provider. The TREC device sends an electronic message when the goods have crossed the border. The message is delivered directly to the Dutch customs office. With our analysis we have demonstrated that the export procedure that involves TREC is an instantiation of the control model generated by the e^3*control* patterns.

The challenge of TREC is in developing a reliable mechanism, to determine which parties are authorized to access a container, at which point in time. This authorization mechanism would require some form of information sharing between the customs’ information systems, and the information systems of participants in a trading process. That means that parties must be carefully scrutinized, before they can join. Such a control by the TREC service providers can be described by the Partner Screening pattern.

The model of controls presented here is used in the ITAIDE project in which businesses and governments, enabled by advanced technology, participate to redesign export procedures. Domain experts from the participating organizations confirmed that the control model does identify real control problems, and suggests a viable solution.

The Proper Contracting and Execution Confirmation patterns have not been described in detail in the Beer Export case study and their examples can only be found within the patterns descriptions. This does not mean that the patterns do not exist in the described network. The reason why we have not modelled them is that the problems that we have been focusing on and which were important to the domain experts do not correspond to these two patterns. Within the beer export network, the problem described by the Proper Contracting pattern could be that the beer producer in the Netherlands does not trust the buyer in the UK and requires him to sign a contract first. This was not considered to be a real-life problem by domain experts. However, theoretically such a situation could exist. For the Execution Confirmation pattern the beer producer in the Netherlands would not trust the buyer in the UK and would require him to send confirmation of delivery. Such confirmation could also be represented by the AAD. However, the problem underlying AAD is not the same that described by the Execution Confirmation pattern.

13.9 Lessons Learned

In the lessons learned we reflect on our expectations about value modelling for controls and compare them to the results of this case study.

Lesson 1. Controls as commercial services with added value. This case study demonstrates that controls are commercial services. This justifies the use of value modelling in the design of controls, as suggested in this thesis.

The first control service that we observe is witnessing and testifying to export. In the AAD and EMCS procedures, Customs NL cannot witness the export of beer outside the Netherlands due to the open EU borders. Therefore, export evidence is provided by Customs UK who acquires it from Retailer UK. The provision of export evidence can be seen as a service, provided by Retailer UK to Customs UK and provided by Customs UK to Customs NL. Because the case study is in the government domain, the service is not provided in return for a monetary reward. Retailer UK and Customs UK are simply obliged to provide export evidence by legislation.

Another service is found in the suggested TREC model. The witnessing of export, provided by the TREC service provider, is the control service. This is a commercial service and someone has to pay for it. In our model we suggest that the service is paid for by the exporter BeerCo NL. However, multiple business models are possible.

One more control service we observe is the Excise Warehouse licensing. Retailers in the UK are licensed to to operate with excise-free shipments. Licensing is a service, provided by customs in the EU, and the retailer pays for it.

Finally, the last control service we observe is certification of TREC service providers. The TREC service providers must be certified to ensure that their technology provides adequate information about export evidence.

Lesson 2. Value aspects of control instruments. The core control instrument in this case study is an evidence document *Export Evidence*. The export evidence is instantiated with the *AAD* document in the AAD procedure, by the *EMCS Record* in the EMCS procedure and by the *TREC Message* in the TREC procedure.

This case study demonstrates that the evidence document *Export Evidence* is of value and, therefore, it is not wrong to model it as a value object. In all the procedures, *Export Evidence* has the role of providing trustworthy information about how much beer was exported to the UK by a Dutch exporter. Having this information is of value to the Customs NL, since only with it can Customs NL make a decision about excise exemptions of the exporter. Therefore, *Export Evidence* and its instances is modelled as a value object.

Other evidence documents presented in this case study are *EW license* and *TREC license*. These are evidence of third party certification. These licenses allow Retailer UK and TREC service providers to be operators of excise-free exports, and therefore, they are of value to them.

We also encounter several penalties and incentives. Firstly, there is a penalty for not paying excises which is modelled as a value object, as suggested by the Penalty pattern. One more penalty discussed in short is the difference in excise tariffs in different EU countries. Such a difference creates a motive for the 'virtual beer' fraud. Making the tariffs equal can be seen as a penalty which removes such an incentive.

Lesson 3. The value viewpoint is an abstract rationale for procedural aspects of controls. This case study demonstrates that modelling controls with value models is possible. Two control problems were identified at the start of the case study: Tampered AAD and Virtual Beer. As a result of the analysis, we have been able to model both problems using the concepts of *e*^{3}*control* .

The *e*^{3}*control* methodology allows us to focus on pure value-related aspects of controls, which provides additional functionality compared to the process perspective on controls. The added value of this functionality has been most strongly demonstrated in the analysis of the TREC service. The advantage of the value modelling when designing the TREC model was that it highlighted the business decisions that had to be made to introduce the TREC solution in practice. The process models only describe how the TREC controls have to be implemented. The value model, on the other hand, pushes the designer into thinking about economic objects, which should be delivered *in return* for the TREC service. In other words, it raises the important question of who will pay for the TREC service.

The analysis of procedural controls along with their value aspects is also important because the choices made in value models should not contradict the control principles. For example, it should be ensured that parties delivering the TREC service do not help exporters to hide excises. Control principles, such as the segregation of duties, should be taken into account when assigning value objects to actors. The case study demonstrates that the *e*^{3}*control* provides tools to design choices, which take into account requirements of both the value and control perspectives.

We therefore conclude that in this case study the combination of process and value perspectives provides a valuable tool to analyze both control and economic value issues related to the introduction of these controls. The analysis performed by *e*^{3}*control* is more powerful than that provided by a value of process perspective separately. The domain experts of the ITAIDE project have also acknowledged that the value perspective provides a simple and relevant view on controls and is particularly useful for the redesign task. As they put it, because the value perspective allows abstracting from procedural detail of old controls, it allows more extreme redesigns compared to those based on the process perspective only.

Lesson 4. Patterns. The controls of the study could be described using the Penalty, Execution Monitoring and Partner Screening patterns. In addition, both delegation patterns have been used.

In general, the patterns produce interesting results. Firstly, they are able to explain the current AAD procedure. Secondly, they provide useful insights into the redesign of this procedure using the TREC technology. Before we carried out this case study it was difficult to make any reasonable arguments with respect how the TREC device would impact the AAD procedure. Some participants on the ITAIDE project had doubts about the ability of the TREC technology to provide sufficient control over the process. By using the patterns, we have analyzed how the introduction of the TREC device will affect the AAD procedure and whether it will improve it. In particular, the controls principles of the patterns were used to demonstrate exactly in which aspects the TREC device can improve the AAD procedure.

In the first e^3*control* cycle we performed an analysis of the AAD procedure. The procedure already includes some elements of the Execution Monitoring pattern. Since our goal was to improve the AAD procedure, the design of the control mechanism was narrowed down to identifying whether the AAD procedure corresponds to the Execution Monitoring pattern, or, more precisely, to its contractual variation Penalty. As a result, by analyzing the compliance of the AAD procedure with the control principles, we are able to identify the weaknesses of the AAD procedure as well as suggest how the weaknesses can be removed. This demonstrates that the patterns can be applied not only to design new controls from scratch, but also to improve existing controls.

In the second e^3*control* cycle we have analyzed the virtual beer fraud. As a result, we used the Partner Screening pattern to explain the necessity of Excise Warehouse licensing. The same pattern explained the necessity of certifying TREC service providers.

The case demonstrates that the application of patterns may not be considered trivial, as was proved in the Virtual Beer scenario, in which we encountered multiple sub-ideal activities and multiple sub-ideally behaving actors. This resulted in the increased possibilities of the controls suggested by the patterns. Sorting out these situations is a topic for future research.

In addition, by applying the Partner Screening pattern to describe EW licensing we encounter different time frames for different activities. Namely, the control activities are executed once per time frame, while the primary activities are executed multiple times. Such a situation is a variation of the original pattern.

We have also managed to create models which comply with the e^3*value* ontology in a sense that they do not contain loops in the dependency paths. Neither do the ideal value models contain one-way value transfers.

13.10 Summary

In this chapter, we have presented the patterns in the case of the excise exemption procedure within the EU. Within the EU, the general principle is that excise is only paid in the country in which excise goods are consumed. Currently, the core evidence of export in the EU is the paper based Administrative Accompanying Document (AAD). In order to export from the Netherlands to the UK, the exporting party sends the document with the cargo to the UK, the document is then stamped in the UK by the local customs authorities and sent back to the Netherlands. Upon presenting the document, the exporter in the Netherlands gets the excise exemption, while the importer pays taxes in the UK.

The AAD procedure does not provide sufficient controls to ensure that excise exemption is granted only for exported goods. There are several ways of avoiding excise payments, e.g. by tampering the AAD document.

In the case study, we have undertaken a re-engineering of the existing AAD procedure and suggested how it could be improved. To find the answer, we have investigated which control problems the AAD procedure tries to cover and how they are addressed by the current AAD procedure. Further, using the e^3*control* patterns, we have explained how the problems can be removed. As a result, we suggest value and process control models, which describe a 'perfect' solution against the control problems in the export scenario.

Furthermore, we suggest a scenario to improve the AAD procedure using new technology. We suggest a scenario to replace the paper-based AAD solution with an electronic one with effective control. This solution is enabled by innovative technologies, such as TREC and EPCIS.

Three patterns were used in this case study: Penalty, Execution Monitoring and Partner Screening. The patterns have proved to be useful. Firstly, Execution Monitoring allows us to identify the problems in the AAD procedure and the Penalty pattern suggests a way of improving it. Both patterns explain why Excise Warehouse licensing is needed and the Partner Screening pattern describes the Excise Warehouse licensing mechanisms.

Finally, the patterns provide guidelines on how to design control procedures using the TREC device. Before we carried out this case study it was not clear if the TREC device would improve the AAD procedure. On the contrary, it could even have been said that the TREC device would provide even fewer controls. By using the patterns, we have analyzed how the introduction of the TREC device will affect the procedure. Based on the theory behind patterns, we conclude that the TREC device, if correctly implemented, will improve controls.

Chapter 14

Conclusions and future research

This chapter presents conclusions and directions for further research on the design of controls in network organizations. We start this chapter with the key outputs of this thesis in section 14.1. In section 14.2 we reflect on our experiences of using e^3*control* with domain experts from the business world. In section 14.3 we discuss managerial implications and, finally, in section 14.4 we outline limitations and directions for future research.

14.1 Key research outputs

The research objective of this thesis was *to develop a methodology that would support human analysts in designing controls against opportunistic behaviour in networks of organizations.* To achieve this, we have introduced the e^3*control* approach that provides guidelines and conceptual modelling tools to design controls in networks.

The e^3*control* approach consists of three components: e^3*control* ontology, e^3*control* patterns and e^3*control* design framework. The e^3*control* ontology provides constructs which are required to describe control problems and control mechanisms from the value perspective. Although the e^3*control* ontology is a necessary part of the whole methodology, our contribution to the ontology development is minor, since an extensive part of it has already been introduced in the e^3*value* ontology. Nevertheless, an important contribution of this research is the approach of modelling controls from the value perspective as a whole, of which the e^3*control* ontology is a part.

Another part of the e^3*control* approach to modelling controls is the use of e^3*control* patterns, which capture *accepted* design knowledge on specific inter-organizational control problems and mechanisms. In this thesis the emphasis has been on the design of controls from the economic value perspective. However, the process perspective is also considered to some extent in e^3*control* patterns.

At the beginning of the thesis we introduced the concept of inter-organizational controls and defined our requirements for the design approach for inter-organizational controls, which, we have argued should incorporate the following features:

- To address the design of controls from the network perspective;
- To address the design of controls from the economic value perspective;
- To provide conceptual modelling techniques which support a human business analyst in the design process. The identified conceptual modelling techniques include *ontologies* and *patterns*.

In order to include these aspects, we have based our design methodology on requirements engineering and internal control theories. We will now go on to discuss how each aspect can be found in the e^3*control* approach.

14.1.1 The network perspective

The controls in this thesis have been designed from the *network perspective*, which contrasts with the traditional practice of the internal control theory in three ways: (1) explicit focus on risks caused by external parties, (2) controls take into account the activities of external parties, and (3) collaborative decision making.

Explicit focus on risks caused by external parties. The internal control theory states that when the controls are introduced, no explicit distinction is made with regard to who causes the risk: an internal employee or an external party. In the network perspective we consider primarily controls against opportunistic behaviour of *counter parties* only. We do not invent new controls: all the control problems and mechanisms, which we consider, have already been described in the internal control literature, agency theory and other sources. Our contribution is to describe these controls explicitly from the network perspective.

The network perspective is most clearly presented in the e^3*control* patterns. These patterns describe pairs of control problems and control mechanisms. Each pattern describes two actors - a primary actor and a counter actor. The counter actor is the one who behaves opportunistically and causes a control problem for the primary actor. The primary actor is the one who wishes to mitigate opportunistic behaviour and is a beneficiary of the control mechanism. This structure corresponds to the network perspective, in which actors are explicitly concerned with the opportunistic behaviour of network partners.

Controls take into account the activities of external parties. Another feature of the network perspective is that the primary actor, who introduces controls, takes into account the activities of other parties in the network. For example, the order of activities between a primary and counter actor is important from the network perspective. This aspect is also incorporated in the patterns which state that a primary actor who has little trust in a counter actor should not execute his commitments before the counter actor does so. Another example of the network perspective in patterns is that when a primary actor delegates activities to a trusted party, it is important to ensure that the party will not act in the interests of the counter actor. In the internal control theory such principles mainly have a purpose to restrict actions of internal employees and are less concerned with opportunistic behaviour of external parties, see in Chapter 2. Our control principles in Chapter 7 represent the 'translation' of the internal control principles for relationships *between organizations*.

The initial effort to design inter-organizational control was made by [Bons, 1997]. Unlike Bons, we did not develop an expert system which uses automated reasoning to select optimal controls for a particular network. Instead the e^3*control* methodology was developed to assist human business analysts in the design of inter-organizational controls. In addition, by integrating the agency theory and the internal control theory, the e^3*control* patterns describe a broader set of controls, than that of Bons. Finally, as will be discussed later, we incorporate the value perspective on controls.

Collaborative decision making. Collaborative decision making is another factor that makes the network perspective different from the traditional internal control theory. For a network to be sustainable, every party has to be content with a business model and, consequently, with the controls incorporated within it. Self-interest seeking partners participate in networks voluntarily, and therefore any agreements in networks, including controls, are typically not decided by one individual manager [Man, 2004]. To achieve sustainability, networks employ collaborative decision making where parties jointly achieve agreements on introduced controls.

To support collaborative decision making, we emphasize the importance of using graphical conceptual modelling techniques when designing controls for networks. Our conceptual models describe processes and value transfers *between* the network parties, and do not just focus on processes *within* one individual company, as in the internal control theory.

14.1.2 The economic value viewpoint on controls

Modelling controls from the *economic value viewpoint* focuses on the way economic value is created, exchanged and consumed in a multi-actor network. At the beginning of the thesis it was argued that the development of the economic value viewpoint of controls has several motives: (1) controls are commercial services with added value, (2) many control instruments have a value aspect (e.g. incentives, tradable evidence documents), and (3) the value viewpoint is an abstract rationale for procedural aspects of controls. We will now go on to discuss whether these arguments have been confirmed in our case studies.

Controls as commercial services with added value. One argument for developing the value viewpoint on controls is the fact that controls often result in commercial services. In all the reported case studies, we were able to observe the controls as being services and we saw how controls changed the business model of the network. In the music case, the control service was the delivery of information on ISRC codes by the listeners to the rights societies. Multiple control services were discernible in the health care case. Examples include needs assessment provided by the indication centre to the ministry, the allocation of actual care services, the allocation of personal budgets provided by the administrative care office to the ministry, the provision of information about care providers, and care provider assessment carried out by the

Social Chart. In the ROC case, we observed the witnessing of green supply, a service that the renewable producer provides to Ofgem. However, this is not a commercial service, since a government organization, Ofgem, simply obliges the producer to provide this information. The Beer Export case also reveals many control services, e.g. providing TREC services, providing the Excise Warehouse service, Excise Warehouse licensing, and signing of the AAD.

This variety of control services discovered in the case studies shows that it is also reasonable to model controls from the economic value viewpoint. When it comes to control services, the value viewpoint has more advantages than the process viewpoint. Firstly, it allows us to demonstrate value-related features of controls without focusing on the procedural details. This is especially useful when designing new control services. For example, we were able to capture the business model of the Social Chart in the health care case without even knowing the details of the businesses processes behind the service. This is very advantageous at the early stage of development of such a service, when it is more important to understand business aspects of the service rather than its procedural aspects. A question such as, 'who is going to pay for the service?' is important. Such decisions should be made at the economic value level and not, in our opinion, be further complicated by involving procedural details.

Value aspects of control instruments. Many control instruments have a value aspect. Examples include incentives/penalties, rights and evidence documents.

Evidence documents as value objects. *Evidence documents* are often associated with economic value; sometimes they can be traded and they have been found in every case study. In the music case, evidence in the form of an ISRC code of a track, is provided by its listener to the rights society. In the health care case study came across many documents including *Delivered Care Evidence*, an evidence document provided by a care provider to the administrative office to confirm how many services were provided. In the ROC case, an evidence document called an ROC certificate is provided by a supplier to Ofgem to show how much green energy has been bought. In the Beer Export case, the *AAD* document, *EMCS record* and *TREC message* are evidence documents which confirm beer export.

We can distinguish two reasons why evidence documents are value objects. The first reason is that they are *of value* to the controlling actors, such as the rights society, an administrative office, Ofgem and customs. These controlling parties cannot perform their core business without having the correct evidence. For example, in the music case a reliable ISRC code is a value object, since without having reliable information about the broadcasted tracks and the number of listeners, right societies cannot accurately charge radio stations *per listener per track*. Similarly, in the Beer Export case, without trustworthy evidence about exported beer, the customs authorities cannot make a correct decision about granting excise exemption.

The second reason why evidence documents can be of value is because they can represent rights to some monetary value and, thus, are tradable. One such document is the cinema ticket demonstrated in Chapter 3. In the ROC case, the ROC certificate represents rights to obtain RO compliance. Therefore, it can be traded.

Modelling evidence documents in the value models allows us to represent the value aspects of the documents. For example, the fact that an ROC certificate represents some value can only be conceptualized in the value models and not in the process models, since the latter do not have a notion of value. Value models also allow us to model the impact of the ROC on a value model, e.g. with e^3*value* we can make a sensitivity analysis of the price and number of the ROCs and, in this way, assess the impact of these parameters on the business model.

Rights as value objects. Another control instrument with a value aspect is a *right*. Rights have been extensively represented by the music and the health care case study. Legal rights can be seen as value objects because they guarantee access to a product or service, and consequently, such rights are of real economic value to actors. For example, a right to AWBZ care in the health care case is of value to people who want access to the AWBZ care services. In the music case, the right to broadcast music is of value to rights owners, since they receive a fee for this right.

The existence of rights implies the existence of objects to which these rights are related. For example, in the music case, we see a value object *Track* as well as a right to broadcast or listen to the track. Modelling of the transfers of rights and transfers of the related value objects in e^3*control* have been extensively studied in Chapter 3 and have been illustrated in the music and the health care case study.

Incentives and penalties as value objects. The case studies contain several incentives and penalties. In the ROC case, there are two penalties and one incentive: the buy-out fees paid by the suppliers for not complying with the renewable obligation, the price of the ROC, which provides an incentive in favour of buying green electricity rather than paying for ROCs, and the buy-out premium for buying green electricity.

In the Beer Export case, we discussed the difference in excise tariffs throughout the Europe. The fact that tariffs are lower in some countries provides an incentive for 'virtual beer' fraud: selling goods in a country with higher excise, but reporting the sale and paying excises in a country with a lower excise. Making excise tariffs equal in all European countries can be seen as a penalty for 'virtual beer' fraud[1].

In the music case, we have suggested a scheme in which information about the number of listeners to a radio station is shared between rights societies and advertisers. Therefore, if a radio station understates the number of listeners, it will pay less rights fees, but it will also receive fewer advertisement fees; this creates an incentive not to understate the number of listeners.

We argue that the properties of incentives and penalties can be better captured by value models rather than by process models. In our view, these presented cases demonstrate that a decision to introduce an incentive with particular properties should be taken at the value level. One of the reasons for this is that the size of the incentive depends on other value transfers. For example, in the ROC case, the size of the buy-out fee depends on the price of regular and renewable electricity (namely, if the buy-out fee is too low, suppliers will prefer to pay the penalty rather than to buy green electricity). The price is a value-level concept and, for purposes of comparison the two, the incentive should also be modelled at the value level.

The value perspective also enables us to capture how an incentive or penalty impacts the profitability of actors. Namely, an incentive increases the value received by a counter actor, while

[1]Making the excise tariffs equal is not the solution offered by the ITAIDE project.

a penalty decreases it. Incentives and penalties could be added to net value sheets in order to assess how the profitability of actors changes with different incentives. This is not possible to model explicitly in process models or information systems models, since they do not have a notion of value.

The value viewpoint is an abstract rationale for procedural aspects of controls. Many controls, encountered in the case study, are procedural. This means that in order to implement them, changes in business processes are required. The value viewpoint has two purposes for procedural controls: (1) to represent a high-level economic motivation of controls and (2) to provide abstraction of procedural details of controls.

A high-level economic motivation of controls. From the value perspective, opportunistic behaviour, such as fraud, embezzlement or just loss of something of value, results in a sub-ideal value transfer and this represents a missing value. If one actor exchanges an ideal value transfer and his counter party exchanges a sub-ideal value transfer, then the first actor provides a value object, but does not get equal value in return. The Principle of Reciprocity is not observed in such cases. A control is needed when the value is missing, i.e. when the principle of reciprocity does not hold. Such an economic motivation of controls is represented with *sub-ideal value models*, using the e^3*control* ontology.

The sub-ideal value model is a tool used to analyze control problems at the value level without knowledge of procedural details. The value perspective is essential in understanding control problems that underlie controls, since the controls are designed primarily to safeguard economic value. An advantage of value modelling for the analysis of controls is that it does not require knowledge of processes. As in the health care case, we were able to describe the control problems without having knowledge of the procedural details.

With respect to other viewpoints, such as a process or information system viewpoint, the sub-ideal value model provides criteria to judge whether the process or information system solution is sufficient to mitigate the identified control problems. As in the music case study, the information system solution does not mitigate all the control problems identified during the value level analysis. A further value-level analysis could be conducted to identify the costs of improving the information system solution and if these costs would be justified by the benefits. This analysis could not be done if it was only limited to the information system perspective.

Abstraction of procedural details of controls. The second purpose of the value viewpoint for procedural controls goes back to the complexity of networks. The design of inter-organizational controls can be very complex, because, for instance, different perspectives of the multiple stakeholders are involved. A way to deal with this multi-perspective problem is to use a multi-viewpoint approach and in this thesis we have focussed on two viewpoints: the economic value viewpoint and the process viewpoint. By introducing a value viewpoint, we provide a high-level model of controls, without focusing on procedural details.

As can be concluded from the case studies, an economic value analysis of control guides the design of the underlying process or information system solution. The latter can be seen as a solution that ensures the occurrence of all value transfers of the corresponding value model. From this point of view, the preliminary value modelling is important, because different economic value solutions require different operational models to implement it. For instance, in the music case study, a business model for paid radio would require a different information system than the business model for free radio considered in our case.

The e^3*control* methodology allows us to focus on pure value-related aspects of control mechanisms, and provides additional functionality compared to the process perspective on controls. In the health care case, we found the economic value perspective important in the design of a new controls service, the Social Chart, which was the most important result for the health care domain experts of the FRUX project [Droes et al., 2005]. Modelling the Social Chart from the economic value perspective yields such issues as who will fund the Social Chart, what service does it provide and who will operate the Social Chart. A further e^3*value* analysis enables us to answer these questions, which is not provided by procedural modelling.

The analysis of procedural controls along with their value aspects is also important because the choices made in value models should not contradict the control principles. The issue of who will operate the Chart is important not only from the value perspective, but also from the control perspective. The operator of the Social Chart should be independent from the care providers or insurance companies, who would like to advertise the service. Similar issues were encountered in the Beer Export case. In the Beer Export case, where the parties who deliver the TREC service should be impartial and not not be prejudiced in favour of a reduction in excise payments.

To summarize these arguments, we conclude that the combination of process and value perspectives provides a valuable tool to analyze both control and economic value issues related to the introduction of these controls. The analysis performed by e^3*control* is more powerful than that provided from a process perspective alone.

14.1.3 The e^3*control* ontology

To design opportunistic behaviour, we have suggested the e^3*control* ontology, which is based on the existing e^3*value* ontology, introduced by [Gordijn and Akkermans, 2003]. The e^3*value* ontology represents concepts to describe a network organization as a set of *actors* that exchange *objects of economic value* with each other, however it does not incorporate control-related aspects, such as opportunistic behaviour. Therefore, we have extended the e^3*value* ontology to specifically include the concepts required to model the effects of opportunistic behaviour on value transfers.

The concepts we have introduced in the e^3*control* ontology include, among other things, *sub-ideal value object* and *sub-ideal value transfer* to show that the result of sub-ideal behaviour is an incorrect exchange of a value object or an exchange of an inappropriate value object. The transfers of sub-ideal value objects are marked in graphical models with *dashed lines*. In addition, we have introduced a *liability token*, a concept that allows us to identify the party who behaves sub-ideally.

As was discovered in the case studies, all the control problems, which we have elicited from domain experts or from literature reviews, could be modelled with the concepts of the e^3*control* ontology. In all cases, the economic motivation behind the control problems could be identified and modelled with the concepts of e^3*control*.

However, e^3*control* concepts were not sufficient when we wished to design more process- or information systems-related aspects of controls. For example, e^3*control* cannot be used to express the sequential order of activities. This means that when procedural or other non-value details become important, an analysis should use the appropriate tools. In our work, some procedural aspects of control are described in the control patterns using UML activity diagrams.

14.1.4 The e^3*control* patterns

In addition to the e^3*control* ontology, the designer also needs to have knowledge about possible control problems and mechanisms that are possible and how they should be implemented in a proper way. This knowledge was structured in e^3*control* patterns, which are defined as descriptions of generic and re-usable control mechanisms for a recurring control problem. In this thesis, we have described four *procedural control patterns* and two *contractual control patterns*.

The procedural patterns include: Partner Screening, Proper Contracting, Execution Monitoring and Execution Confirmation. The Partner Screening and Execution Monitoring patterns describe monitoring-based mechanisms to combat a counter actor who executes his contracting obligations sub-ideally. The difference between the two patterns is that Partner Screening suggests an ex-ante monitoring control, and Execution Monitoring suggests an ex-post monitoring control. Further, the Proper Contracting pattern describes how the contracting process should be arranged to avoid misunderstandings about the contracting commitments the actors make with each other. Finally, the Execution Confirmation pattern suggests how the final process of a transaction should be arranged to avoid misunderstandings about the performed activities. The control mechanisms described in the procedural patterns affect only the process viewpoint, unless a control requires some activity to be delegated. Delegation creates a service, modelled at the value level as a value transaction. In such a case, the control changes the initial value model. Delegation is modelled by combining procedural patterns with special *delegation patterns*.

Furthermore, there are two contractual patterns: Penalty and Incentive. Both patterns describe an actor's choice of behaving ideally or sub-ideally. The Penalty pattern describes a mechanism of punishing sub-ideal behaviour. The Incentive pattern describes a mechanism of rewarding ideal behaviour. In these patterns the control mechanism affects both the value and the process viewpoint.

The patterns have been demonstrated in detail in two case studies: the ROC case and the Beer Export case. In the ROC case we describe the Execution Monitoring, the Incentive and the Penalty patterns. The Beer Export case includes the Execution Monitoring and Partner Screening patterns. Other patterns, although not demonstrated, could also be found in the case studies.

The patterns can also be observed in the other two case studies. For instance, in the music case, the rights societies monitor the number of listeners and tracks played by the radio station. This ex-post monitoring process can be described by the Execution Monitoring pattern. The witnessing of the number of listeners and tracks is done by the listeners, who report this information to the rights societies. The information is then sent by the listener to the rights societies in a seamless model so that the listener does not interact with the process.

In the health care case, several patterns can be found as well. For example, the government agency CVZ monitors the performance of care providers to ensure that patients get good quality care. This corresponds to the controls described in the Execution Monitoring pattern. Another scenario is when a care provider may claim to have no commitment to provide care for a given budget, and hence refuses to continue to provide care. In accordance with the Proper Contracting pattern, the government only commits a specific budget, if a care provider promises to provide the agreed care for that budget. The patterns in the health care case are analyzed in more detail in [Kartseva et al., 2007b].

We have not invented the controls described in the patterns. We have structured existing knowledge of controls by employing the patterns approach. The structuring of the knowledge of patterns is a contribution in itself, since, in our view, the controls in the internal control theory are not structured in an efficient way. Typical descriptions of internal controls are very domain-specific. Firstly, separate risks and controls are described for every specific activity of each transaction cycle. Secondly, they assume the presence of some specific documents or parts of IS (e.g. picking tickets, packing slips, bar codes, and credit card processing systems). As a result, a control designer must be acquainted with a huge number of control descriptions. For example, in the controls described in [Romney and Steinbart, 2006], a control expert, who wants to design controls for the sales and procurement processes inside an organization should have thorough knowledge of about one hundred controls.

In the patterns, we describe the controls at a higher level of abstraction, without a specific focus on a business process or an industry. This is possible, since we start by modelling the control problems with domain-independent value models and initially describe a control problem as a sub-ideal value transfer. Thus, the controls described in the patterns do not depend on a specific business process, but only on a structure of the value transfers. As a result, we are able to describe controls using a rather small number of patterns. In the case studies, we are able to describe all the encountered control problems and mechanisms with the six patterns and their delegation variations. We consider a limited number of patterns covering a large number of cases as a contribution.

Another contribution of the patterns, as was already argued, is that they describe the control problems and control mechanisms in a rigorous way using graphical conceptual models. Furthermore, the patterns describe not only the business process viewpoint, but also the economic value viewpoint of each control problem and control mechanism. Finally, the patterns specifically describe controls from the network perspective.

14.2 Reflection on experience with the *e^{3}control* approach

The *e^{3}control* approach has been applied in four case studies. In every case study the *e^{3}control* modelling has been done by the author of this thesis. A case study was usually performed as follows; firstly, we elicited the knowledge about existing control problems and existing control mechanisms. This was done by interviewing domain experts, participating in meetings with them, or studying documents, including the materials of earlier case studies. Secondly, we carried out the *e^{3}control* modelling of the control problems and mechanisms ourselves. As much as possible, the *e^{3}control* models were presented to, corrected by and discussed with the domain experts. If this was not possible, the models were verified with the available documented sources. For more information about this process see the Research Context section of the chapter of each specific case study.

Our general impression is that conceptual thinking and understanding of simple diagrams in general was not new to the domain experts who, generally speaking, had had no *e*^{3}*control* training. It is worth mentioning that the domain experts who worked with the *e*^{3}*value* and *e*^{3}*control* models had differing and non-IT backgrounds, ranging from medical experts in the health care case to customs officers in the Beer Export case. In our experience, many domain experts tended to use some kind of graphical conceptual diagram to explain their ideas. For example, in the health care case one interviewee at the indication centre CIZ drew her own "conceptual" diagram (not an *e*^{3}*value* one), to explain how the Dutch health care system works.

In the ITAIDE project, the graphical conceptual models of *e*^{3}*value* in particular were used during the redesign workshops with domain experts. The goal of the joint workshops was to come up with a redesign of current customs procedures. The whole communication and decision making process concerning redesign was based on the *e*^{3}*control* and process models, in part presented here in the Beer Export case study. The models were extensively used and discussed with the domain experts. They were presented in a Power Point presentation and discussed within the group. The domain experts would give their comments on the model, ask why this or that value transfer was needed and what it represented, and then suggested their own ideas on redesign. With the use of *e*^{3}*control* models, the ideas generated in the workshops were implemented in the real-life demonstrator, which consisted of ten TREC devices fitted on pilot containers shipping beer to various countries within and outside of the EU.

However, in our opinion, in some cases the *e*^{3}*control* models contained too many details for the domain experts. The most intuitively understood concepts are the actor, the value object, and the value transfer. This being the case, we presented simplified e3controlmodels to the domain experts in some cases. We did not usually present the dependency path and we omitted some value objects which were not relevant to the topic of discussion.

The fact that we did not show some concept like the dependency path does not mean that these concepts are useless. Every concept in *e*^{3}*control* is needed to perform a rigorous analysis of the *e*^{3}*control* models by the *e*^{3}*control* expert. However, our claim is that simplified models may serve a better purpose when discussed with domain experts who are not familiar with *e*^{3}*control* .

In the ROC case, we also experimented with teaching the domain experts to perform the *e*^{3}*value* (not *e*^{3}*control*) modelling by themselves. This was done within the EU-funded project BusMod. For this, we organized a two day workshop, during which all the concepts of *e*^{3}*value* were explained in detail to a group of around ten domain experts. After that the experts were asked to build their own *e*^{3}*value* models with the use of a workbook, in which each modelling step was explained in detail and examples were provided. The experts were also given assistance by *e*^{3}*value* experts. In the months following the workshop, the experts would perform the analysis of a more complex case on their own. Our experience has been that most of the domain experts were able to build and analyze the models. However, they still needed some assistance from *e*^{3}*value* experts even after the workshop. This assistance was given by e-mail or phone, and in two out of the four cases the *e*^{3}*value* expert would visit the domain expert's site for a couple of days to assists with the modelling. A typical occurrence that was discovered during the BusMod project is that the experts tended to model process-related details in the e^3*value* models [Kartseva et al., 2004b].

The conclusion we come to from this experience, additional e^3*value* training does not guarantee that domain experts who have no experience with conceptual modelling in general will be able to do the modelling on their own, and certainly not on at expert level. Therefore, in the interests of efficiency, it is better to let domain experts carry out the analysis and design of controls together with business analysts trained in e^3*value* . The optimal way of using e^3*control* is to allow a domain expert, who has knowledge of conceptual modelling, to do the analysis, with the proviso that he or she should be trained in the e^3*control* methodology.

To sum up, we recommend the following to domain experts who wish to use the e^3*control* but have not had the appropriate training in it:

- The e^3*control* analysis should be performed jointly by a domain expert(s) without an e^3*control* training and a business analyst trained in e^3*control* (i.e. the e^3*control* expert). All e^3*control* modelling should be done by the e^3*control* expert.
- The e^3*control* expert should elicit knowledge from the domain expert and perform several iterations of the e^3*control* cycle (ideal model – sub-ideal model – solution) by improving the models through feedback from the domain expert.
- A more simplified version of the e^3*control* models can be presented to domain experts unfamiliar with the e^3*control* ontology. The level of simplification should depend on the purpose of the presentation. For instance, if in focus, value transfers should be modelled in detail, whereas the dependency path can be omitted, if not in focus.

14.3 Managerial implications

In addition to the theoretical contributions described, this study has provided new insights for practical business management. This study has illustrated the importance of adopting a network perspective in the context of control. It has been argued that an understanding of the business model of the network as a whole enables us to take the sustainability of the network into account when implementing controls. This kind of network view is specifically needed in relation to the kind of managerial challenges faced by a company operating as a part of a network organization. As this study has illustrated, it is not enough for a company to concentrate on developing and sharpening internal controls, but it also needs to understand and try to take into account or even influence the design of controls in the entire network. The broadening of the scope of control of a company's internal processes towards external issues is therefore an important managerial challenge for a company operating in a network environment.

The study has also emphasized the importance of assessing the value impact of controls. It has been argued that understanding the value aspects of control mechanisms is crucial, since it allows us to make more rational decisions about implementing controls. The e^3*control* tools can be used for a cost-benefit analysis of controls.

14.4 Research limitations and future research

As with all research, there are limitations to the interpretation of the results and other issues that need to be considered when trying to generalize this analysis to broader issues of interest. Some limitations as well as the related directions for future research are discussed in this section.

A limited view on the notion on control. In the start of the study, we assumed the definition of control to be that of the internal control theory, which implies a somewhat limited view on control problems and mechanisms. For example, in this study inter-organizational controls are seen here as measures to mitigate opportunistic behaviour. However, in the control literature, a second objective of controls, which is frequently put forward, is that controls also should be designed in such a way as to motivate partners to reach the alliance's objectives efficiently and effectively [Geringer and Hebert, 1989], [Groot and Merchant, 2000]. This objective has not been considered in this work.

Un addition, there are other theories, in which the controls are studied from a different perspective and with different purposes. Attention is paid to various *factors* that influence the selection of one or another control mechanism. For example, Transaction Costs Economics [Williamson, 1979] studies how control mechanisms are influenced by such factors as uncertainty and asset specificity. They state that the degree of coordination in an inter-organizational relation increases with the degree of interdependence and uncertainty of tasks performed by parties in this relationship. Thus, this theory provides rules for selecting a *level of control* depending on other properties of the network.

We have not incorporated rules for the selection of a certain level of control in the e^3*control* methodology because we assume that users of the methodology will make such a selection for themselves, e.g. based on a case-specific knowledge. The only distinction we provide is a difference between high trust and low trust between the network partners. In the case of high trust, the controls can be relaxed. The introduction of a more sophisticated model to select the levels of control would be a good extension of the methodology and could be a topic of future research.

Furthermore, e^3*control* is not an extensive risk management assessment methodology, as is for example the COSO methodology in auditing. In general, we have described control problems based on a case and we do not go into an analysis of what the probability is that a risk materializes. Extending the methodology with functionality, which provides guidelines on the selection of certain controls or on assessment of severity levels of risks, can be a good extension to the methodology.

Generalizability and Completeness . External validity deals with knowing whether the results are generalizable beyond the immediate case. As was explained earlier in Chapter 1, generalizability cannot be guaranteed in the four case studies.

Nevertheless, we claim that the e^3*control* is generalizable for the set of problems and mechanisms similar to those described by the case studies. For instance, we are quite sure that tax payment fraud can be described by using the e^3*control* patterns, since a similar problem with excise payment fraud has already been described. In general, all four case studies have a strong regulatory flavour. Many of the control problems which have come to light are the result of a violation of some global governmental regulation or a contractual agreement. In our opinion, *e^3control* is capable of designing problems and mechanisms similar to those described in the case studies.

It is questionable whether the list of *e^3control* patterns is complete in the sense that it covers all the possible control problems and mechanisms in network organizations. In this research, the patterns were primarily elicited based on certain literature sources, as described in Chapter 6. Certainly, we cannot claim that these patterns describe all possible control problems and mechanisms. However, in the case studies we have performed, all the control problems we discovered could be described by the patterns. We have a tentative personal impression that most of the relevant control mechanisms have been identified, but there is no empirical proof of this.

Therefore, there is no guarantee that a complete and generalisable set of the patterns has been described. The patterns library can be enriched by carrying out new case studies, finding other literature streams, or focusing on a specific business domain.

Focus on value and process views. The *e^3control* patterns describe the control problems and mechanisms only from the value and process viewpoints. Other viewpoints, such as the information system viewpoint or the communication viewpoint, described by [Bons, 1997] have not been taken into account. Such problems and control mechanisms are described, for example, by security patterns, which are concerned with solutions for recurring information security problems [Schumacher, 2003]. Other problems can be related to accuracy and completeness of data and information exchanged in messages between organizations. We do not consider these viewpoints and related problems within *e^3control* . An important extension of the *e^3control* patterns in particular would be to describe the correspondence of the information system-level problems and solutions to their value-level descriptions.

Cost-benefit analysis of controls. The *e^3value* methodology enables the financial valuation of profitability of each actor, a functionality which has not been used in this thesis. By making such a financial analysis on the sub-ideal value models and comparing the results with the results of the ideal value models, it is possible to calculate how much value is lost due to sub-ideal behaviour or how much value is gained in an illegal way, as in the Beer Export case. We have not performed such financial valuations of the cases, since our focus was on the conceptual development of the methodology. However, such valuation is essential in order to completely grasp the suggested solutions and extend the understanding of control problems.

Aggregation of multiple controls. Multiple control problems in networks are possible. Firstly, every actor in a network has different perspectives on what he thinks a control problem is. Even in a two-actor relationship, each of two actors can play the role of the primary actor, and consider the other actor to behave sub-ideally. In a three-actor network, there are even more control problems, since each of the three actors can be seen as a primary actor, who considers the other two actors to behave sub-ideally.

The assumption of the e^3*control* patterns, which deal with this issue, is that controls in a network can be represented as an aggregation of controls between the pairs of actors in this network. In this respect, in the e^3*control* patterns we only consider the control problems caused by a counter actor to a primary actor. Even in a two-actor network, the control mechanisms Actor A introduces to control Actor B should be aggregated with the mechanisms Actor B introduces against Actor A. In a three-actor network, the control can be described as an aggregation of controls of Actor A , Actor B, and Actor C. The idea behind this is to handle the complexity of control problems in multi-actor networks by splitting them into parts of problems considered by every single actor. There are several problems with such an approach.

The controls for Actor A and controls for Actor B may contradict. Such a case was reported in [Kartseva et al., 2006e], where the controls of the seller require the delivery to be executed before the payment, while the controls of the buyer require the opposite. The solution to this conflict reported in [Kartseva et al., 2006e] is the use of the escrow service. Other similar mechanisms, which solve such conflicts have not been investigated in this study. The resolution of such conflicts is a topic of future research.

Feature interaction. Another problem related to the aggregation of multiple controls is that, after aggregation, the outcome of the final solution is defined by the order of introducing the controls. The solution when we first introduce controls for Actor A and then for Actor B will most probably differ from the solution when we introduce the controls in the reverse order, first those for Actor B and then those for Actor A. In conceptual modelling this phenomenon is known as *feature interaction*.

This phenomena is also relevant for the e^3*control* patterns. In an exchange between a buyer and a seller different controls will be introduced depending on whether we start with the buyer or the seller. As an example, in the Execution Monitoring pattern, if we start from the point of view of the buyer, we will require the seller to deliver goods before payment. However, if we start as the seller, we impose the prepayment requirement for the buyer. So, we impose different controls depending on what actor's perspective we take first.

A more structured approach is needed to resolve the problems of feature interaction. Examination of the effects of feature interaction in the e^3*control* patterns and development of corresponding design rules is the topic of future research.

14.5 Future of the design of controls in networks

During the recent years many books have been published on network organizations [Tapscott et al., 2000], [Weill and Vitale, 2001], [Man, 2004], [Seuring et al., 2003]. Many books on control mechanisms have seen the light either [Hollander et al., 1999], [Romney and Steinbart, 2006], [Starreveld et al., 1994]. In general, the design of networks and controls are seen as two distinct tasks, where the design of networks is more focused on efficiency issues rather than on controls. What is lacking is a truly multi-disciplinary approach to the design of controls in networks. The e^3*control* approach takes a step towards such an approach.

The design of controls in networks is a relevant issue nowadays and many developments in current society and businesses require new, effective solutions for inter-organizational controls. A good example is developments in international supply chains. On the one hand, growing threats, such as those to security and health require increased control of goods going through international supply chains. On the other hand, growing volumes of trade make it impossible to exert extensive control inspections at borders, and the administrative burden of businesses should be lowered in order to create and maintain a viable economic zone. However, efficiency and a reduced administrative burden can easily contradict increased security, safety and control. For example, the threat of terrorism has resulted in new control regulations.

Similar challenges are also faced by other industries such as the food industry. On the one hand, it is essential to be able to trace the origin of food in order to stop contaminations or the spread of deceases in case such dangers arise. On the other hand, the market requires quicker and more flexible delivery and distribution systems.

New control procedures are required in order to cope with these challenges. Designing and implementing changes in controls in network organizations is a highly complex issue, where technological, financial and political stakes have to be aligned. It obviously requires decisions on value aspects, since new business models will emerge. In spite of the challenges, new technologies such as Internet and sensor-based tracking technologies create new opportunities for the design of lean and effective controls. For example, RFID technologies are being introduced to improve tracking systems in the food industry.

In addition, the field of control in networks is experiencing a major paradigm shift. A good example is the idea of implementing of Authorized Economic Operators (AEO) in EU custom procedures, which implies that each EU Member State Customs administration establishes a partnership with the private sector in its country in order to involve the private sector in ensuring the safety and security of international trade supply chains. Certified AEOs will enjoy tangible benefits such as fast customs clearance and simplified procedures. Such partnerships between government and private sectors is a fundamental shift in paradigm of control. While in the past customs control was considered to be an issue for only customs authorities, nowadays businesses are seen as partners, and a win-win situation is required, such that businesses are responsible for the control of their own supply chains, and customs can rely on this control. Because this relieves customs administrations from control tasks, these businesses can be rewarded with simplifications of procedures [Rukanova et al., 2006].

To summarize, there are several important developments in the field. Firstly, societal changes such as globalisation, an increase in the volume of international trade, and terrorism require more controls, but fewer administrative burdens. Secondly, technology provides new ways to achieve these objectives. Finally, there is a paradigm shift, at least within the EU, of outsourcing of control by the government to the private sphere. This dynamic setting makes a future research in the design of inter-organizational controls to be a relevant, exciting and promising field of research.

Appendices

Appendix A

Controls principles

Table A.1: Control principles of pattern ‘Partner Screening’

Principle ID	**Control principles**
	Activities-Activities principles
V-Ia	Witness activity must exist
V-II	Verify activity must exist
P-I	Promise activity must exist
V-IIIa	Witness activity must follow Past Counter activity
V-IVa	Verify activity must follow Witness activity
PE-I	Verify activity must precede Primary activity
P-II	Promise activity must precede Primary activity
	Activities-Objects principles
V-V	Supporting statement must be directly transferred to Verify activity from a source that generates it
V-VIa	To-be-verified statement must be directly transferred from Witness activity to Verify activity
P-II	Confirmation statement must be directly transferred from the primary actor’s Promise activity to the counter actor
	Activities-Actors principles
V-VIIa	An actor executing Witness activity must be independent and socially detached from the actor executing Counter activity.
V-VIII	An actor executing Verify activity must be independent and socially detached from the actor executing Counter activity
V-IX	Supporting statement must generated by an actor independent and socially detached from the actor executing Counter activity
V-VIIa	An actor executing Witness activity must be independent and socially detached from the actor executing Past Counter activity.

V-VIII	An actor executing Verify activity must be independent and socially detached from the actor executing Past Counter activity
V-IX	Supporting must generated by an actor independent and socially detached from the actor executing Past Counter activity

Table A.2: Control principles of pattern 'Proper Contracting'

Principle ID	**Control principles**
	Activities-Activities principles
V-II	Verify activity must exist
V-IIIb	Promise(CA) must precede Verify activity
P-I	The Verify activity must precede the primary actor's Promise(PA) activity
PE-I	Promise(CA) activity must exist
PE-II	The counter actor's Promise(CA) activity must always precede the primary actor's Promise(PA) activity
	Activities-Objects principles
V-V	Supporting must be directly transferred to Verify activity from a source that generates it
V-VIb	Confirmation(CA) must be directly transferred to Verify activity
P-III	Confirmation(PA) statement of the primary actor must be directly transferred from the Promise(PA) activity to the actor execution Verify.
PE-I	Promise(PA) activity of the primary actor must precede Primary activity.
	Activities-Actors principles
V-VIII	An actor executing Verify activity must be independent and socially detached from the actor executing Promise(CA) activity.
V-IX	Supporting must generated by an actor independent and socially detached from the actor executing Promise(CA) activity.

Table A.3: Control principles of pattern 'Execution Monitoring'

Principle ID	**Control principles**
	Activities-Activities principles
V-Ia	Witness activity must exist
V-II	Verify activity must exist

V-IIIa	Witness activity must follow Counter activity
V-IVa	Verify activity must follow Witness activity
PE-I	Verify activity must precede Primary activity
	Activities-Objects principles
V-V	Supporting statement must be directly transferred to Verify activity from a source that generates it
V-VIa	To-be-verified statement must be directly transferred from Witness activity to Verify activity
	Activities-Actors principles
V-VIIa	An actor executing Witness activity must be independent and socially detached from the actor executing Counter activity.
V-VIII	An actor executing Verify activity must be independent and socially detached from the actor executing Counter activity
V-IX	Supporting statement must generated by an actor independent and socially detached from the actor executing Counter activity

Table A.4: Control principles of pattern 'Execution Confirmation'

Principle ID	**Control principles**
	Activities-Activities principles
V-Ia	Witness activity must exist
V-II	Verify activity must exist
V-IIIa	Witness activity must be executed at the same time as Primary activity
V-IVa	Verify activity must follow Witness activity
T - I	Testify activity must exist
T - II	Testify activity must follow Primary Activity
PE - I	Verify activity must precede the Testify activity
	Activities-Objects principles
V-V	Supporting statement must be directly transferred to Verify activity from a source that generates it
V-VIa	To-be-verified statement must be directly transferred from Witness activity to Verify activity
T-III	The outcome of the Verify activity (Positive statement or Negative Statement, both not shown in the model) must be directly transferred from Verify activity to Testify activity
T-IV	Testifying Statement must be directly transferred from Testify to the primary actor
	Activities-Actors principles

V-VIIa	An actor executing Witness activity must be independent and socially detached from the actor executing Counter activity.
V-VIII	An actor executing Verify activity must be independent and socially detached from the actor executing Counter activity
V-IX	Supporting statement must generated by an actor independent and socially detached from the actor executing Counter activity
T-V	Testify activity must be executed by the same actor who receives the Positive Statement from the Verify activity

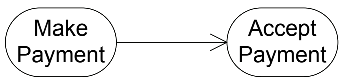
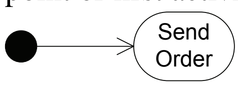
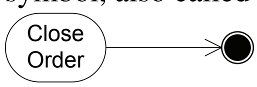
Appendix B

UML Activity Diagrams

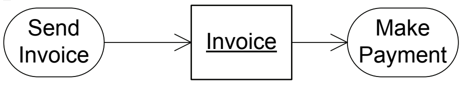
In UML, an activity diagram is used to display the sequence of activities [Fowler and Scott, 2000]. Activity diagrams show the workflow from a start point to the finish point detailing the many decision paths that exist in the progression of events contained in the activity. They may be used to detail situations where parallel processing may occur in the execution of some activities. Activity diagrams are useful for business modelling where they are used for detailing the processes involved in business activities. In Table B.1 we describe elements of UML activity diagrams.

Table B.1: Elements of an UML activity diagram

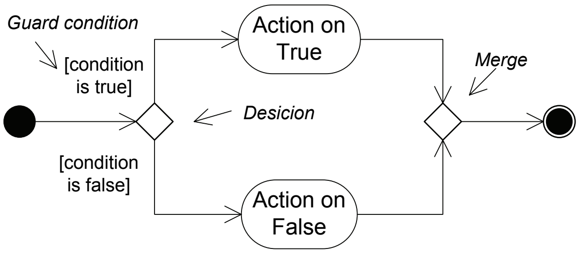
<table>
<tr><th>Element and its description</th></tr>
<tr><td>Activity: An activity is the specification of a parameterized sequence of behaviour. An activity is shown as a round-cornered rectangle enclosing all the actions, control flows and other elements that make up the activity.

</td></tr>
<tr><td>Control Flow: A control flow shows the flow of control from one action to the next. Its notation is a line with an arrowhead.

</td></tr>
<tr><td>Initial Activity: Initial activity, also called initial node, shows the starting point or first activity of the flow. It is depicted by a large black spot.

</td></tr>
<tr><td>Final Activity: The end of the Activity diagram is shown by a bull's eye symbol, also called as a final activity or final node.

</td></tr>
</table>

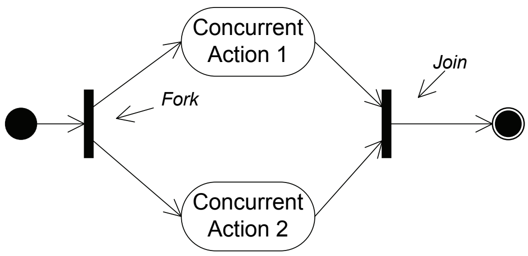
Objects and Object Flows: An object flow is a path along which objects or data can pass. An object is shown as a rectangle. An object flow is shown as a connector with an arrowhead denoting the direction the object is being passed.



Decision and Merge Nodes: Decision nodes and merge nodes have the same notation: a diamond shape. The control flows coming away from a decision node will have guard conditions which will allow control to flow if the guard condition is met. A merge passes any control flows straight through it. If two or more inflows are received by a merge symbol, the action pointed to by its outflow is executed two or more times.



Fork and Join Nodes: Forks and joins have the same notation: either a horizontal or vertical *synchronization bar* (the orientation is dependent on whether the control flow is running left to right or top to bottom). They indicate the start and end of parallel threads of control. The join synchronizes two inflows and produces a single outflow. The outflow from a join cannot execute until all inflows have been received.



Swim lanes: An activity partition is shown as either a horizontal or vertical swim lane. In the following diagram, the partitions are used to separate actions within an activity into those performed by a seller and those performed by a buyer.

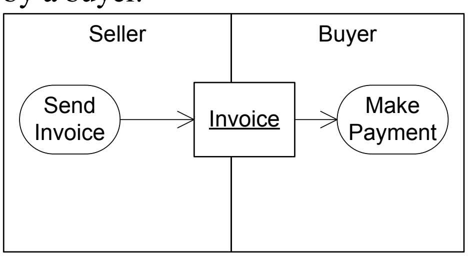


Figure B.1 shows an example of a UML activity diagram for an ordering process. The model starts with the seller who executes the activity Receive Order, and follows with the decision fork, where the seller makes a decision to accept the order or not. In the first case, the seller ships the order and, in parallel, sends an invoice. The buyer pays the invoice. After the order is shipped and the payment is accepted, the seller closes the order. If the order was not accepted at all, it is also closed, which is modelled by the merge node before the activity Close Order.

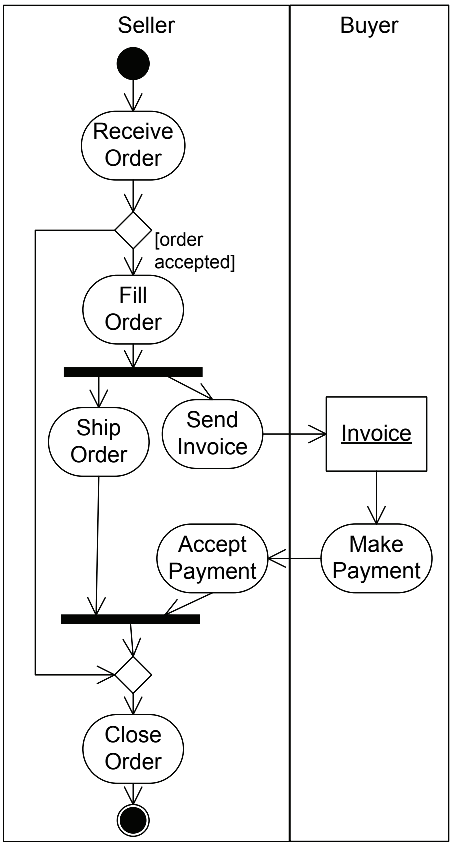


Figure B.1: Example of the UML activity diagram

Bibliography

[ACFE, 2006] ACFE (2006). Report to the nation on occupational fraud and abuse. Technical report, Association of Certified Fraud Examiners (ACFE).

[Akkermans and Gordijn, 2006] Akkermans, H. and Gordijn, J. (2006). What is this science called requirements engineering? In *RE '06: Proceedings of the 14th IEEE International Requirements Engineering Conference (RE'06)*, pages 266–271, Washington, DC, USA. IEEE Computer Society.

[Alchian and Demsetz, 1972] Alchian, A. A. and Demsetz, H. (1972). The property right paradigm. *The Journal of Economic History*, 33(1):16–27.

[Alexander, 1979] Alexander, C. (1979). *The Timeless Way of Building*. Oxford, Oxford University Press.

[Alstyne, 1997] Alstyne, M. V. (1997). The state of network organization: A survey in three frameworks. *Journal of Organizational Computing*, 7.

[Andersson et al., 2006] Andersson, B., Bergholtz, M., Edirisuriya, A., Ilayperuma, T., Johannesson, P., Grgoire, B., Schmitt, M., Dubois, E., Abels, S., , Hahn, A., Gordijn, J., Weigand, H., and Wangler, B. (2006). Towards a reference ontology for business models. In *Proceedings of 25th International Conference on Conceptual Modeling(ER2006)*, Tucson, Arizona, USA.

[Anthony and Govindarajan, 2003] Anthony, R. N. and Govindarajan, V. (2003). *Management Control Systems*. McGraw Hill Higher Education, 11rev ed edition edition.

[Arrow, 1985] Arrow, K. J. (1985). *The Economics of Agency*, chapter Principals and Agents: The Structure of Business, pages 37–51. Harvard Business School Press, Boston, MA.

[Austin, 1962] Austin, J. (1962). *How to DO things with words*. Harvard University Press, Cambridge, MA.

[Baida, 2006] Baida, Z. (2006). *Software-aided Service Bundling - Intelligent Methods and Tools for Graphical Service Modeling*. PhD thesis, Vrije Universiteit, Amsterdam, NL.

[Baida et al., 2007] Baida, Z., Rukanova, B., Liu, J., and Tan, Y.-H. (2007). Rethinking eu trade procedures - the beer living lab. In *Proceedings of The 20th Bled eCommerce Conference (Bled 2007)*, Bled, Slovenia.

[Bakos, 1998] Bakos, Y. (1998). The emerging role of electronic marketplaces on the internet. *Communications of the ACM*, 41(8):35 42.

[Bekkers et al., 2002] Bekkers, R., Duysters, G., and Verspagen, B. (2002). Intellectual property rights, strategic technology agreements and market structure - the case of gsm. *Research Policy*, 31(7):pp. 1141–1161.

[Bons et al., 2000] Bons, R. W., Dignum, F., Lee, R. M., and Tan, Y.-H. (2000). A formal analysis of auditing principles for electronic trade procedures. *International Journal of Electronic Commerce*, 5(1):57–82.

[Bons et al., 1998] Bons, R. W., Lee, R. M., and Wagenaar, R. W. (1998). Designing trustworthy interorganizational trade procedures for open electronic commerce. *International Journal of Electronic Commerce*, 2(3):61–83.

[Bons, 1997] Bons, R. W. H. (1997). *Designing Trustworthy Trade Procedures for open Electronic Commerce*. PhD thesis, University of Rotterdam.

[Borgida et al., 1982] Borgida, A., Mylopoulos, J., and Wong, H. K. (1982). *On Conceptual Modelling*, chapter Generalization/Specification as a Basis for Software Specification, pages 87–119. Spribger-Verlag.

[Borst et al., 1997] Borst, P., Akkermans, H., and Top, J. (1997). Engineering ontologies. *International Journal of Human-Computer Studies*, 46:365406.

[Brathwaite, 1980] Brathwaite, K. S. (1980). *Applications Development Using CASE Tools*. Academic Press Inc., San Diego, Ca.

[Brussels, 2006] Brussels (2006). Eu coherent strategy against fiscal fraud frequently asked questions (memo/06/221). Internet.

[Buhr, 1998] Buhr, R. J. A. (1998). Use-case maps as architectural entities for complex systems. *IEEE Transactions on Software Engineering*, 24(12):1131–1155.

[Campbell, 2006] Campbell, D. E. (2006). *Incentives: motivation and the economics of information*. Cambridge University Press, 2 edition.

[Chen and Lee, 1992] Chen, K. and Lee, R. (1992). Schematic evaluation of internal accounting control systems. EURIDIS Research Monograph, RM-1992-08-1, Erasmus University Rotterdam.

[Coad, 1992] Coad, P. (1992). Object-oriented patterns. *Communications of the ACM*, 35(9):152–159.

[Coase, 1960] Coase, R. H. (1960). The problem of social cost. *Journal of Law and Economics*, 3:1–44.

[Coplien and Schmidt, 1995] Coplien, J. and Schmidt, D., editors (1995). *Pattern Languages of Program Design*. Addison Wesley, Reading, MA.

[Coplien and Harrison, 2004] Coplien, J. O. and Harrison, N. (2004). *Organizational Patterns of Agile Software Development*. Prentice Hall.

[COSO, 2004] COSO (2004). Enterprise risk management integrated framework. Technical report, Commitee of Sponsoring Organisations, American Institute of Certified Public Accountants, Jersey City.

[Cross, 2001] Cross, N. (2001). *Design knowing and learning: cognition in design education*, chapter Design cognition: results from protocol and other empirical studies of design activity, page 79103. Elsevier, Amsterdam.

[Dannecker and Lechner, 2006] Dannecker, A. and Lechner, U. (2006). Success factors of communities of patients. Gteborg.

[Das and Teng, 1998] Das, T. K. and Teng, B.-S. (1998). Between trust and control: Developing confidence in partner cooperation in alliances. *The Academy of Management Review*, 23(3):491–511.

[Dekker, 2004] Dekker, H. C. (2004). Control of inter-organizational relationships: Evidence on appropriation concerns and cordnation requirements. *Accounting, Organisations and Society*, 29:27–49.

[Droes et al., 2005] Droes, R., Meiland, F., van der Roest, H., Doruff, C., Varodi, I., Akkermans, J., Baida, Z., Faber, E., Haaker, T., Moelaert, F., Kartseva, V., and Tan, Y. (2005). A dynamic interactive social chart in dementia care; attuning demand and supply for patients and carers. In *Proceedings of the 12th Congress of the International Psycho-geriatric Association*, Stockholm.

[Dyer and Singh, 1998] Dyer, J. H. and Singh, H. (1998). The relational view: Cooperative strategy and sources of interorganizational competitive advantage. *Academy of Management Review*, 23(4):660–679.

[Eisenhardt, 1989] Eisenhardt, K. M. (1989). Agency theory: An assessment and review. *Academy of Management Review*, 14(1):57.

[Exter et al., 2004] Exter, A., Hermans, H., Dosljak, M., and Busse, R. (2004). Health care systems in transition: Netherlands. Technical report, WHO Regional Office for Europe, Copenhagen.

[Finkelstein et al., 1994] Finkelstein, A., Gabbay, D., Hunter, A., and Nuseibeh, B. (1994). Inconsistency handling in multi-perspective specifications. *IEEE Transactions on Software Engineering*, 20(8):569–578.

[Finkelstein et al., 1992] Finkelstein, A., Kramer, J., Nuseibeh, B., Finkelstein, L., and Goedicke, M. (1992). Viewpoints: A framework for integrating multiple perspectives in system development. *International Journal of Software Engineering and Knowledge Engineering*, 2(1):31–58.

[Fowler, 1997] Fowler, M. (1997). *Analysis Patterns: Reusable Object Models*. Addison-Wesley, Reading, MA.

[Fowler and Scott, 2000] Fowler, M. and Scott, K. (2000). *UML Distilled*. Addison Wesley, 2nd edition edition.

[Gambetta, 1988] Gambetta, D. (1988). *Trust*, chapter Can we trust trust?, pages 213–237. Basil Blackwell, New York.

[Gamma et al., 1995] Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1995). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison Wesley, Boston.

[Geringer and Hebert, 1989] Geringer, J. and Hebert, L. (1989). Control and performance of international joint ventures. *Journal of International Business Studies*, pages 235–254.

[Gordijn, 2002] Gordijn, J. (2002). *Value-based requirements Engineering: Exploring innovatie e-commerce ideas*. PhD thesis, Vrije Universiteit Amsterdam.

[Gordijn and Akkermans, 2001] Gordijn, J. and Akkermans, J. (2001). *e3-value*: Design and evaluation of e-business models. *IEEE Intelligent Systems, Special Issue on e-Business*, 16(4):11–17.

[Gordijn and Akkermans, 2003] Gordijn, J. and Akkermans, J. (2003). Value-based requirements engineering: Exploring innovative e-commerce ideas. *Requirements Engineering*, 8(2):114–134.

[Groot and Merchant, 2000] Groot, T. and Merchant, K. (2000). Control of international joint ventures. *Accounting, Organizations and Society*, 25:579–607.

[Gruber, 1994] Gruber, T. R. (1994). *Formal Ontology in Conceptual Analysis and Knowledge Representation*, chapter Towards principles for the design of ontologies used for knowledge sharing. Amsterdam, NL.

[Hanpachern, 1982] Hanpachern, A. (1982). System for detecting fades in television signals to delete commercials from recorded television. United States Patent Number 4319286.

[Hayes et al., 2005] Hayes, R., Dassen, R., and Wallage, A. S. P. (2005). *Principles of Auditing, an introduction to international standards on auditing*. Prentice Hall-Financial Times, Harlow, 2 edition.

[Hevner et al., 2004] Hevner, A. R., Ram, S., and March, S. T. (2004). Design science in information systems research. *Mangement Information Systems Quarterly*, 28(1):75–105.

[Holbrook, 1999] Holbrook, M. (1999). *Consumer value - A Framework for analysis and research*. Routledge, New York.

[Hollander et al., 1999] Hollander, A., Denna, E., and Cherrington, J. (1999). *Accounting, Information Technology, and Business Solutions*. McGraw-Hill, Irwin, 2 edition.

[Houlder, 2007] Houlder, V. (2007). Surge in vat fraud costs states up to 100bn. The Financial Times.

[ISACF, 2005] ISACF (2005). Control objectives for information and related technology 4.0 (cobit 4.0). Technical report, Information Systems Audit and Control Foundation (ISACF).

[Jensen and Meckling, 1976] Jensen, M. C. and Meckling, W. H. (1976). Theory of the firm: Managerial behavior, agency costs and ownership structure. *Journal of Financial Economics*, 3(4):305–360.

[Jones and Porn, 1985] Jones, A. J. and Porn, I. (1985). Ideality, sub-ideality and deontic logic. *Synthese*, 65(2):275–290.

[Kartseva et al., 2004a] Kartseva, V., Gordijn, J., and Tan, Y.-H. (2004a). Analysing preventative and detective control mechanisms in international trade using value modeling. In *Proceedings of the Sixth International Conference on Electronic Commerce (ICEC'04)*.

[Kartseva et al., 2004b] Kartseva, V., Gordijn, J., and Tan, Y.-H. (2004b). Value-based business modelling for network organizations: Lessons learned from the electricity sector. In *Proceedings of the 12th European Conference on Information Systems (ECIS'04), Turku*.

[Kartseva et al., 2005a] Kartseva, V., Gordijn, J., and Tan, Y.-H. (2005a). Designing control mechanisms for value webs: The internet radio case study. In Vogel, D. R., Walden, P., Gricar, J., and Lenart, G., editors, *Proceedings of the 18th Bled e-comerce conference*, Bled, Slovenia.

[Kartseva et al., 2005b] Kartseva, V., Gordijn, J., and Tan, Y.-H. (2005b). Developing a modelling tool for designing control mechanisms in network organisations. *International Journal of Electronic Commerce*.

[Kartseva et al., 2006a] Kartseva, V., Gordijn, J., and Tan, Y.-H. (2006a). Inter-organisational controls as value objects in network organisations. In *CAiSE*, pages 336–350.

[Kartseva et al., 2006b] Kartseva, V., Hulstijn, J., Baida, Z., Gordij, J., and Tan, Y.-H. (2006b). Towards control patterns for business networks. In Vervest, P. and van Heck, E., editors, *Proceedings of the Smart Business Network Initiative Discovery Event*, Putten, Netherlands. Springer.

[Kartseva et al., 2006c] Kartseva, V., Hulstijn, J., Gordij, J., and yao Hua Tan (2006c). Modelling value-based inter-organizational controls in healthcare regulations. In *Proceedings of the 6th IFIP conference on e-Commerce, e-Business, and e-Government*, Turku, Finland. IEEE.

[Kartseva et al., 2007a] Kartseva, V., Hulstijn, J., Gordijn, J., and Tan, Y.-H. (2007a). Control patterns in a healthcare network. In *Proceedings of the 8th Annual International Conference on Digital Government Research (DGo'2007)*.

[Kartseva et al., 2007b] Kartseva, V., Hulstijn, J., and Gordijn, Y.-H. T. J. (2007b). Control patterns in a healthcare network. In *Proceedings of the 8th Annual International Conference on Digital Government Research (DGo'2007)*, Philadelphia, PA.

[Kartseva et al., 2005c] Kartseva, V., Hulstijn, J., Tan, Y.-H., and Gordijn, J. (2005c). Health care control and governance: Modeling health care control mechanisms and discovering business opportunities for information provision and quality assurance. Technical report, FREEBAND.

[Kartseva et al., 2006d] Kartseva, V., Hulstijn, J., Tan, Y.-H., and Gordijn, J. (2006d). Control patterns in a healthcare network: Applying control patterns to redesign governance and control mechanisms in a healthcare network. Technical report, FREEBAND.

[Kartseva et al., 2006e] Kartseva, V., Hulstijn, J., Tan, Y.-H., and Gordijn, J. (2006]e). Towards value-based design patterns for inter-organizational control. In ?, editor, *Proceedings of the 19th Bled Electronic Commerce Conference on eValues, Bled, Slovenia, June 2006*. University of Maribor.

[Kartseva and Tan, 2005] Kartseva, V. and Tan, Y.-H. (2005). Designing controls for a marketplace of health care services: a case study. In *Proceedings of the 12th Research Symposium on Emerging Electronic Markets (RSEEM 2005)*.

[Keil, 2005] Keil, P. (2005). Principal agent theory and its application to analyze outsourcing of software development. In *Proceedings of the International Workshop on Economics-Driven Software Engineering Research (EDSER)*, St. Louis, USA. IEEE Computer Society.

[Knechel, 2001] Knechel, W. (2001). *Auditing: Assurance and Risk*. South-Western College Publishing, Cincinnati, Ohio, 2 edition.

[Kotler, 1988] Kotler, P. (1988). *Marketing Management: Analysis, Planning, Implementation, and Control*. Prentice Hall, Englewood Cliffs, N.J.

[Laresgoiti et al., 2004] Laresgoiti, I., Madina, C., Pedrosa, L., and Diaz, A. (2004). D 4.4 energy services case study. Technical report, EESD Project NNE5/2001/256 BUSMOD.

[Loucopoulos and Karakostas, 1995] Loucopoulos, P. and Karakostas, V. (1995). *System Requirements Engineering*. McGraw-Hill, Berkshire, UK.

[Malone, 1987] Malone, T. (1987). Modeling coordination in organizations and markets. *Management Science*, 33(10):1317–1332.

[Malone et al., 1999] Malone, T., Crowston, K., Lee, J., Pentland, B., Dellarocas, C., Wyner, G., Quimby, J., Osborn, C., Bernstein, A., Herman, G., Klein, M., and Donnell, E. (1999). Tools for inventing organizations: Toward a handbook of organizational processes. *Management Science*, 45(3).

[Man, 2004] Man, A.-P. D. (2004). *Network Economy: Strategy, Structure And Management*. Edward Elgar Publishing Inc.

[Manns, 2001] Manns, M. L. (2001). Patterns: a promising approach to knowledge management. In *First Annual ABIT Conference*, Pittsburgh, Pennsylvania.

[March and Smith, 1995] March, S. T. and Smith, G. F. (1995). Design and natural science research on information technology. *Decis. Support Syst.*, 15(4):251–266.

[Martin et al., 1998] Martin, R., Riehle, D., and Buschmann, F., editors (1998). *Pattern Languages of Program Design 3*. Addison-Wesley, Reading, MA.

[Mayer et al., 1995] Mayer, R., Davis, J., and Schoorman, F. (1995). An integrative model of organizational trust. *Academy of Management Review*, 20(3):709–734.

[McCarthy, 1982] McCarthy, W. E. (1982). The REA accounting model: A generalized framework for accounting systems in a shared data environment. *The Accounting Review*, pages 554–578.

[Merchant and van der Stede, 2007] Merchant, K. and van der Stede (2007). *Management Control Systems*. Prentice Hall, Harlow, 2 edition.

[Morris et al., 2005] Morris, M., Schindehutte, M., and Allen, J. (2005). The entrepreneur's business model: Toward a unified perspective. *Journal of Business Research*, 58(6):726–735.

[Mutale and Strbac, 2004] Mutale, J. and Strbac, G. (2004). Task 4.2 distributed generation case study scenarios applying the e3value business modeling. a case study of active management of distribution networks with distributed generation in the uk. Technical report, EESD Project NNE5/2001/256 BUSMOD.

[Noy and McGuinness, 2001] Noy, N. F. and McGuinness, D. L. (2001). Ontology development 101: A guide to creating your first ontology. Technical report, Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880.

[NPD, 2007] NPD (2007). Legal music downloads were fastest growing digital music category in 2006. Press Release.

[OFGEM, 2004a] OFGEM (2004a). Renewables Obligation - Atlantic Electric and Gas Limited and Maverick Energy Limited. Technical Report R/65, Office of Gas and Electricity Markets, United Kingdom.

[OFGEM, 2004b] OFGEM (2004b). The renewables obligation: Annual report 2003-2004. Technical report, Office of Gas and Electricity Markets, United Kingdom.

[OFGEM, 2006] OFGEM (2006). Renewables obligation: Third annual report, ref: 35/06. Technical report, Office of Gas and Electricity Markets, United Kingdom.

[Okma, 2001] Okma, K. (2001). Health care, health policies and health care reforms in the netherlands. Technical report, Ministerie van Volksgezondheid, Welzijn en Sport.

[Osterwalder and Pigneur, 2002] Osterwalder, A. and Pigneur, Y. (2002). An ebusiness model ontology for modeling ebusiness. In *Proceedings of the 15th Bled Electronic Commerce Conference – eReality: Constructing the eEconomy (Bled'02)*, pages 75–91.

[Ouchi, 1979] Ouchi, W. (1979). A conceptual framework for the design of organizational control systems. *Management Science*, 25(9):833–848.

[Pateli and Giaglis, 2004] Pateli, A. and Giaglis, G. (2004). A research framework for analysing business models. *European Journal of Information Systems*, 13(4):302–304.

[Pedrinaci et al., 2005] Pedrinaci, C., Baida, Z., Akkermans, H., Bernaras, A., Gordijn, J., and Smithers, T. (2005). Music rights clearance: Business analysis and delivery. In *Proceedings of the 6th International Conference on Electronic Commerce and Web Technologies*, volume 3590 of *Lecture Notes in Computer Science (LNCS)*, pages 198–207, Copenhagen, Denmark. Springer-Verlag.

[Porter and McKibbon, 1988] Porter, L. W. and McKibbon, L. (1988). *Management Education and Development: Rift or Thrust into the 21st Century?* McGraw-Hill, New York, NY.

[Porter, 1980] Porter, M. E. (1980). *Competetive Strategy. Techniques for analyzing industries and competitors.* The Free Press, New York, NY.

[Rolland et al., 2000] Rolland, C., Stirna, J., Prekas, N., Loucopoulos, P., Persson, A., and Grosz, G. (2000). Evaluating a pattern approach as an aid for the development of organisational knowledge: An empirical study. In Wangler, B. and Bergman, L., editors, *Lecture Notes in Computer Science*, volume 1789, pages 176–191. Springer Berlin / Heidelberg.

[Romney and Steinbart, 2006] Romney, M. and Steinbart, P. (2006). *Accounting Information Systems*. Prentice Hall, New Jersey, 10th edition.

[Ross, 1973] Ross, S. (1973). The economic theory of agency: The principal's problem. *The American Economic Review*, 63(2):134–139.

[Rukanova et al., 2006] Rukanova, B., Koldijk, F., van Ipenburg, F., Dekker, G., Felix, J., Voogel, J., Liu, J., Bezemer, N., Wessels, R., Haazelager, R., Visscher, W., Tan, Y.-H., and Baida, Z. (2006). Beer living lab report on initial site survey and problem definition. Deliverable no. d5.1 of information technology for adoption and intelligent design for e-government (itaide) project. project no. 027829, Vrije Universiteit Amsterdam.

[Rumbaugh et al., 1999] Rumbaugh, J., Jacobson, I., and Booch, G. (1999). *The Unified Modelling Language Reference Manual*. Addison Wesley Longman, Reading, MA,.

[Schonberger, 1986] Schonberger, R. (1986). *World Class Manufacturing*. The Free Press, New York, NY.

[Schubert and Ginsburg, 2000] Schubert, P. and Ginsburg, M. (2000). Virtual communities of transaction: Personalization in electronic commerce. *Electronic Markets*, 10(1):45–55.

[Schumacher, 2003] Schumacher, M. (2003). *LNCS*, volume 2754. Springer-Verlag, Berlin Heidelberg.

[Seruca and Loucopoulos, 2003] Seruca, I. and Loucopoulos, P. (2003). Towards a systematic approach to the capture of patterns within a business domain. *The Journal of Systems and Software*, (67):1–18.

[Seuring et al., 2003] Seuring, S., Mller, M., Goldbach, M., and Schneidewind, U., editors (2003). *Strategy and Organization in Supply Chains*. Physica-Verlag Heidelberg.

[Simon, 1976] Simon, H. A. (1976). *Administrative behavior a study of decision-making processes in administrative organization*. The Free Press, New York.

[Simons, 2000] Simons, R. (2000). *Performance Measurement and Control Systems for Implementing Strategy*. Prentice Hall, Upper Saddle River, New Jersey.

[Sobrero and Schrader, 1998] Sobrero, M. and Schrader, S. (1998). Structuring inter-firm relationships: a meta-analytic approach. *Organization Studies*, 19(585615).

[Sommerville and Sawyer, 1997] Sommerville, I. and Sawyer, P. (1997). *Requirements Engineering: A Good Practice Guide*. John Wiley & Sons, Inc., New York, NY, USA.

[Spence, 1973] Spence, M. (1973). Job market signaling. *Quarterly Journal of Economics*, (88):355–74.

[Starreveld et al., 1994] Starreveld, R., de Mare, B., and Joels, E. (1994). *Bestuurlijke Informatieverzorging (in Dutch)*, volume 1. Samsom, Alphen aan den Rijn, 4th edition.

[Susman and Evered, 1978] Susman, G. and Evered, R. D. (1978). An assessment of the scientific merits of action research. *Administrative Quarterly*, 23:582–603.

[Tan and Thoen, 2000] Tan, Y.-H. and Thoen, W. (2000). Towards a generic model of trust for electronic commerce. *International Journal of Electronic Commerce*, 5(2):61–74.

[Tapscott et al., 2000] Tapscott, D., Lowy, A., and Ticoll, D. (2000). *Harnessing the Power of Business Webs*. Harvard Business School Press, Boston, MA.

[Uschold et al., 1998] Uschold, M., King, M., Moralee, S., and Zorgios, Y. (1998). The enterprise ontology. *The Knowledge Engineering Review*, 13(1):31–89.

[van Aken, 2004] van Aken, J. E. (2004). Management research based on the paradigm of the design sciences: The quest for field-tested and grounded technological rules. *Journal of Management Studies*, 41(2):219–246.

[Varian, 2006] Varian, H. L. (2006). *Intermediate Microeconomics: A Modern Approach*. W.W. Norton & Company, 7 edition.

[Weigand and de Moor, 2003] Weigand, H. and de Moor, A. (2003). Workflow analysis with communication norms. *Data and Knowledge Engineering*, 47(3):349–369.

[Weigand et al., 2006] Weigand, H., Johannesson, P., Andersson, B., Bergholtz, M., Edirisuriya, A., and Ilayperuma, T. (2006). On the notion of value object. In *CAiSE*, pages 321–335.

[Weill and Vitale, 2001] Weill, P. and Vitale, M. (2001). *Place to Space : Migrating to eBusiness ModelsEconomy and Society*. Harvard Business School Press, Boston:, MA, 2001 edition.

[Wieringa et al., 2003] Wieringa, R., Blanken, H., Fokkinga, M., and Grefen, P. (2003). Aligning application architecture to the business context. In Eder, J. and Missikoff, M., editors, *Proceedings of the Conference on Advanced Information System Engineering (CAISE 2003)*, LNCS 2681, page 209225, Berlin Heidelberg. Springer-Verlag.

[Wieringa et al., 2005] Wieringa, R., Maiden, N., Mead, N., and Rolland, C. (2005). Requirements engineering paper classification and evaluation criteria: a proposal and a discussion. *Requir. Eng.*, 11(1):102–107.

[Wieringa and Meyer, 1993] Wieringa, R. J. and Meyer, J.-J. C. (1993). *Applications of Deontic Logic in Computer Science: A Concise Overview*, pages 17–40.

[Williamson, 1979] Williamson, O. E. (1979). Transaction cost economics: The governance of contractual relations. *Journal of Law and Economics*, 22:3–61.

[Witte, 1972] Witte, E. (1972). Field research on complex decision-making processesthe phase theorem. *Int Stud Manage Organ*, (2):156 – 182.

[Yin, 1994] Yin, R. (1994). *Case study research: Design and methods*. Sage Publishing, Beverly Hills, CA, 2 edition.

[Yu and Mylopoulos, 1998] Yu, E. and Mylopoulos, J. (1998). Why goal-oriented requirements engineering. In Dubois, E., Opdahl, A., and Pohl, K., editors, *Proceedings of the Fourth International Workshop on Requirements Engineering: Foundations of Software Quality*, pages 15–22, Pisa, Italy. Presses Universitaires de Namur, Paris.

Samenvatting

Het ontwerpen van controlemechanismen in netwerkorganisaties vanuit een waardeperspectief

Tegenwoordig worden diensten steeds vaker door een netwerk van partijen aangeboden. Als een consument bijvoorbeeld een andere ringtone voor zijn mobiele telefoon wil hebben, heeft de consument met al snel meerdere partijen te maken. Fabrikanten van mobiele telefoons moeten allereerst hun telefoon op zodanig manier ontwerpen dat de ringtone gemakkelijk kan worden aangepast. Tenslotte zijn er aanbieders van de ringtones zelf nodig, en die aanbieders moeten rekening houden met de (on)mogelijkheden van mobiele telefoons. Verder moeten die aanbieders betaald worden voor iedere verkochte ringtone; de betaling verloopt vaak via de mobieletelefonie operator.

Al deze partijen samen vormen een netwerk. Steeds meer organisaties kiezen bewust voor deze vorm van samenwerking: de netwerkstrategie. Door in een netwerk te participeren kunnen netwerkorganisaties aan een complexe klantbehoefte voldoen, waaraan deze organisaties zelfstandig nooit zouden kunnen voldoen.

De netwerkstrategie zien we veel in *e-commerce* en *e-business* toepassingen, waar informatietechnologie zoals het Internet en Web technologie een intrinsiek en belangrijk onderdeel vormen van de propositie naar de consument. Een netwerkstrategie vereist dat organisaties kennis en informatie met elkaar delen en activiteiten met elkaar coordineren. Organisaties zullen dit echter alleen doen als er voldoende onderling vertrouwen is. Dit vertrouwen is niet altijd aanwezig. Daarom worden er vaak controlemechanismen in het netwerk ingebouwd die het mogelijk maken opportunistisch gedrag (zoals fraude of het niet nakomen van afspraken) in het netwerk tijdig te detecteren of zelfs dergelijk gedrag te voorkomen. In het geval van een netwerkorganisatie spelen interorganisationele controlemechanismen een belangrijke rol. Onder *interorganisationele controlemechanismen* verstaan wij de door een partij opgestelde lijst van maatregelen met als doel het opportunistische gedrag van een tegenpartij te voorkomen, te detecteren en/of te corrigeren.

Waarden en controle. De focus van dit proefschrift is gericht op de analyse en het ontwerp van controlemechanismen in netwerkorganisaties, beschouwd vanuit het economisch waarde perspectief. In netwerkorganisaties zullen partijen objecten van economische waarde met elkaar gaan uitwisselen, zoals diensten, goederen, en geld. Deze waarde-uitwisselingen vormen een belangrijk aandachtspunt wanneer we opportunistisch gedrag van partijen beschouwen; immers opportunistisch gedrag van organisaties zal zich juist veelal richten op deze waarde-uitwisselingen.

In dit proefschrift ontwikkelen we een ontwerpmethodologie, de zogenaamde e^3*control* methodologie, die ondersteuning biedt bij het ontwerpproces van controlemechanismen. In deze methodologie staat het begrip economische waarde centraal. Met economische waarde bedoelen wij de waarde (bijvoorbeeld, in geld uitgedrukt), die toegekend kan worden aan producten of diensten door een partij die ze wil kopen. Controlemechanismen kunnen zelf ook als commerciële diensten beschouwd worden, als deze diensten een bepaalde waarde hebben voor de partijen in een netwerk. Men kan bijvoorbeeld een waarde toekennen aan een controlemechanisme dat administratieve fraude kan detecteren of voorkomen. Met het toekennen van waarden kan een oordeel worden gegeven over het effect van controlemechanismen op het businessmodel van het netwerk.

In de e^3*control* methodologie worden controlemechanismen vanuit drie perspectieven geanalyseerd: het economische waarde-uitwisseling perspectief, het bedrijfsprocessen perspectief waarmee waarde-uitwisselingen gerealiseerd worden en het informatiesysteem perspectief waarmee de waardenuitwisselingen en bedrijfsprocessen worden uitgevoerd.

Dit onderzoek is gebaseerd op twee theoretische pijlers: enerzijds de bedrijfswetenschappelijke accounting- en controletheorie, en anderzijds de informatiekundetheorie. Controlemechanismen zijn uitvoerig onderzocht in de accounting- en controletheorie. Dergelijk onderzoek is echter hoofdzakelijk beperkt tot interne bedrijfsprocessen binnen een organisatie en wordt niet specifiek op netwerkorganisaties toegepast. De focus in de beroepspraktijk van accounting en controle is voornamelijk gericht op het onderzoeken van bedrijfsprocessen en informatietechnologie binnen een organisatie. De e^3*control* methodologie voegt hieraan de waardenanalyse van interorganisationele controlemechanismen tussen bedrijven in netwerkorganisaties toe.

In de informatiekunde zijn allerlei methodes ontwikkeld voor het preciezer, formeler beschrijven en analyseren van processen en organisaties. Zulke methodes zijn gebaseerd op het idee van *conceptueel modelleren*. De informatiekunde heeft de conceptuele modellen voor bedrijfsprocessen en informatiesystemen uitvoerig bestudeerd. Het opstellen van dergelijke modellen om software requirements te ontwikkelen in samenspraak met betrokken partijen wordt ook wel *requirements engineering* genoemd. In de accountancy worden dergelijke conceptuele modellen ook wel voor het analyseren en ontwerpen van controlemechanismen gebruikt.

Een specifieke methode om waarde-uitwisselingen te modelleren is e^3*value*. Deze methode is echter nog niet toegepast op het ontwikkelen van controlemechanismen in netwerkorganisaties. In dit onderzoek ontwikkelen we een methode die juist ondersteuning biedt bij het ontwerpen van controlemechanismen met gebruikmaking van een economische waardemodel. De e^3*control* methodologie is gebaseerd op de e^3*value* methodologie met daaraan toegevoegd speciale modelleringsconcepten voor het modelleren van controlemechanismen.

De e^3*control* methodologie. De e^3*control* methodologie bestaat uit drie onderdelen. In dit proefschrift worden de drie onderdelen beschreven, namelijk:

1. Een raamwerk dat aangeeft hoe een netwerkorganisatie geanalyseerd kan worden met betrekking tot het ontwerpen van interorganisationele controlemechanismen;

2. Een ontologie die een kader aanreikt voor het ontwerp en analyse;

3. Een verzameling patronen van controleproblemen en -oplossingen, die als bouwstenen gebruikt kunnen worden in het ontwerpproces van controlemechanismen.

1. Het raamwerk. Het e^3*control* raamwerk beschrijft het ontwerpproces om tot controlemechanismen te komen in drie stappen. De eerste stap behelst de beschrijving van een *ideale situatie* waarin wordt verondersteld dat alle partijen in het netwerk de regels van het netwerk volgen. De tweede stap identificeert mogelijke *controleproblemen*, dat wil zeggen situaties die kunnen ontstaan als sommige partijen in het netwerk misbruik maken van andere partijen en handelen zoals beschreven in de ideale organisatie, en representeert deze sub-ideale toestand in een *sub-ideaal waardemodel*. De derde stap ontwerpt *controlemechanismen* die de controleproblemen zoals gevonden in stap 2 kunnen verminderen of zelfs wegnemen.

2. Het waardemodel: de ontologie. Het ideale waardemodel wordt middels een e^3*value* model beschreven. De e^3*control* benadering voegt concepten toe om ook een sub-ideaal waardemodel te kunnen beschrijven. Deze uitbreiding noemen we de e^3*control* ontologie. De e^3*control* ontologie is afgeleid van de e^3*value* ontologie en bevat voor een deel dezelfde concepten zoals *actor*, *waardeobject*, *waardenuitwisseling* en *dependencypad*.

Een actor vertegenwoordigt een organisatie (of bedrijf) in het netwerk. Waardeobjecten omvatten goederen en diensten die de bedrijven met elkaar uitwisselen. Waardeobjecten zijn ook de financiële vergoedingen die de bedrijven betalen voor de uit te wisselen goederen en/of diensten. Met dependencypaden worden de verschillende afhankelijkheden tussen de waardeobjecten gemodelleerd.

De belangrijkste toevoeging van de e^3*control* ontologie ten opzichte van e^3*value* is het onderscheid tussen ideale en sub-ideale situaties. In ideale situaties is er sprake van actoren, die zich volgens de regel van het netwerk gedragen, die de ideale objecten van waarden uitwisselen en bovendien een ideaal dependencypad volgen. In sub-ideale situaties gedragen de actoren zich niet zoals is voorgeschreven. Er is dan sprake van *sub-ideale waardenuitwisselingen* en *sub-ideale dependencypaden*. Een andere toevoeging is het *boete* concept, waarmee het sub-ideale gedrag wordt toegerekend aan een specifieke actor in het netwerk.

3. Controle patronen. Het derde deel van de e^3*control* methode bestaat uit e^3*control* patronen. Patronen zijn beschrijvingen van controlemechanismen die nuttig kunnen zijn in netwerken. De beschrijving is voor het grootste deel afkomstig uit de accounting en controle literatuur, maar is aangepast voor netwerkorganisaties. Sommige patronen zijn afkomstig uit case studies die tijdens dit onderzoek zijn uitgevoerd. Enkele voorbeeldpatronen zijn het vooraf controleren van de betrouwbaarheid en geschiktheid van een tegenpartij, en het geven van een beloning die voorkomt dat een tegenpartij zich onbetrouwbaar gedraagt.

Hoewel de primaire focus van dit proefschrift gericht is op het modelleren van controles met waardenmodellen, was het voor de patronen ook noodzakelijk de gerelateerde bedrijfsprocessen te beschrijven. Dat was vooral nodig omdat waardemodellen niet de volgordelijkheid van waardeuitwisselingen beschrijven, en deze volgordelijkheid soms wel van belang is voor het begrip van de controlemechanismen.

Hoewel het modelleren van de processen noodzakelijk is, is veel informatie over controlemechanismen al wel af te leiden uit de waardenmodellering. Denk daarbij aan controlemechanismen zoals de uitwisseling van eigendomsrechten, de uitwisseling van bewijsdocumenten en vooral het toekennen van effectieve boetes en beloningen bij het analyseren van de waardendistributie in het netwerk. Toch is de relatie tussen bedrijfsprocessen en waardemodellen een onderwerp dat nog nader onderzocht moet worden.

Case studies. De bruikbaarheid van de e^3*control* methodologie is onderzocht in vier case studies op het gebied van regelgeving in de digitale muziek, gezondheidszorg, elektriciteit en internationale handel. In alle case studies hebben wij concrete controle problemen geanalyseerd en controlemechanismen daarvoor ontwikkeld. Hierbij beschrijven we de case studies in het kort.

Internet radio De Internet radio case is gericht op het correct verdelen van geinde vergoedingen voor het uitzenden van muziek via het Internet. Voor het uitzenden van muziek is een radio-omroep verplicht de makers van muziek een vergoeding te betalen. Er zijn op het moment van schrijven geen betrouwbare procedures om precies te bepalen hoeveel mensen naar een muziekstuk hebben geluisterd en hoeveel keer het nummer is uitgezonden. Hierdoor is het niet mogelijk om de makers van de muziek exact te betalen waar ze recht op hebben. In deze case studie hebben wij controleprocedures ontwikkeld die, door middel van moderne technologie (zoals digitale versleuteling), het mogelijk maken om op een betrouwbare manier de muziekrechten in kaart te brengen en hiermee uiteindelijk de juiste vergoedingen te bepalen.

De Nederlandse gezondheidszorg In deze case hebben wij het complexe proces van publieke gezondheidszorgverzekering in Nederland bestudeerd. In het bijzonder hebben we de ‘Algemene Wet Bijzondere Ziektekosten’ (AWBZ) onderzocht. Het huidige gezondheidsverzekeringsysteem bevat vele processen die niet optimaal functioneren. Bijvoorbeeld: ouderen en chronisch zieke mensen krijgen weinig informatie over mogelijkheden om alternatieve zorg (zoals verzorging door een familielid) in te schakelen en daarvoor een vergoeding te ontvangen. We hebben onderzocht in hoeverre de e^3*control* methodologie in staat was deze inefficiënties (in een sector die niet op winst is georiënteerd) in kaart te brengen en te beschrijven. In de casestudie hebben wij een voorstel gemaakt om de bestaande problemen op te lossen.

Duurzame energie In deze case studie hebben we onderzoek gedaan naar de ontwikkeling van een beloning- en boetesysteem voor het stimuleren van de productie in duurzame energie, zoals windenergie en zonne-energie voor de energiesector. In deze casestudie is specifiek gekeken naar de wet ‘Duurzame Verplichting’ die in Engeland van kracht is. Deze wet verplicht elektriciteitsleveranciers om een bepaald percentage (bijv. 10%)

van de te leveren elektriciteit te laten bestaan uit duurzame elektriciteit. Leveranciers van duurzame energie (bijvoorbeeld beheerders van een windmolenpark) kunnen zogenaamde 'Duurzame Verplichting Certificaten' geven aan elektriciteitsleveranciers. De elektriciteitsleveranciers gebruiken deze certificaten om hun verplichte duurzame bijdrage te kunnen verantwoorden aan de overheid. Door een gebrek aan duurzame energiebronnen stijgt de prijs van een certificaat, met als gevolg dat de energiemarkt wordt gestimuleerd om nieuwe duurzame productie-installaties te bouwen. In deze case hebben wij controlepatronen gebruikt om bestaande en nieuwe controlemechanismen voor het uitwisselen van certificaten te ontwikkelen, te controleren en te optimaliseren.

Export van bier In deze case is onderzoek gedaan naar de accijnsprocedure binnen de Europese Unie. Als een accijnsgoed, zoals bier, verkocht wordt in het land waar het geproduceerd is, dan moet de producent daarover accijns betalen aan de belastingdienst. Als het bier echter in een ander land wordt verkocht dan waar het geproduceerd is, dan hoeft de producent geen accijns te betalen. In dit geval betaalt de koper in het land van invoer de accijnskosten. Het innen van de accijns is een complex systeem en heeft veel zwakke plekken, waardoor fraude mogelijk is. Door middel van het toepassen van de e^3*control* methodologie op de bestaande situatie, hebben wij onderzocht hoe dit systeem kon worden verbeterd. Tevens hebben wij gekeken naar de vraag hoe de papieren documenten, die nu worden gebruikt voor fraude preventie, konden worden vervangen door elektronische gegevensuitwisseling om de fraudegevoeligheid te beperken.

Conclusie. Dit onderzoek heeft aangetoond dat het mogelijk is om controleproblemen en controlemechanismen te analyseren door middel van waardenmodellen. Hoewel het niet met zekerheid te stellen is dat de beschreven controlemechanismen (zoals ze in de controlepatronen staan) voor de in dit proefschrift onderzochte industrieën volledig zijn, hebben de case studies wel aangetoond dat de controlepatronen gebruikt kunnen worden om een uitgebreide verzameling van controle aspecten te beschrijven. Op basis van de uitkomsten van de case studies zijn we tot de conclusie gekomen dat de e^3*control* methodologie, en in het bijzonder de controlepatronen daarin, bruikbaar is voor het analyseren en ontwerpen van controlemechanismen voor netwerkorganisaties.

Tinbergen Institute Dissertation Series

The Tinbergen Institute is the Institute for Economic Research, which was founded in 1987 by the Faculties of Economics and Econometrics of the Erasmus Universiteit Rotterdam, Universiteit van Amsterdam and Vrije Universiteit Amsterdam. The Institute is named after the late Professor Jan Tinbergen, Dutch Nobel Prize laureate in economics in 1969. The Tinbergen Institute is located in Amsterdam and Rotterdam. The following books recently appeared in the Tinbergen Institute Research Series:

372. A. HOPFENSITZ, The role of affect in reciprocity and risk taking: Experimental studies of economic behavior.
373. R.A. SPARROW, Health, education and economic crisis: Protecting the poor in Indonesia.
374. M.J. KOETSE, Determinants of investment behaviour: Methods and applications of meta-analysis.
375. G. MILLER, On the role of personality traits and social skills in adult economic attainment.
376. E.H.B. FEIJEN, The influence of powerful firms on financial markets.
377. J.W. GROSSER, Voting in the laboratory.
378. M.R.E. BRONS, Meta-analytical studies in transport economics: Methodology and applications.
379. L.F. HOOGERHEIDE, Essays on neural network sampling methods and instrumental variables.
380. M. DE GRAAF-ZIJL, Economic and social consequences of temporary employment.
381. O.A.C. VAN HEMERT, Dynamic investor decisions.
382. Z. SASOVOVA, Liking and disliking: The dynamic effects of social networks during a large-scale information system implementation.
383. P. RODENBURG, The construction of instruments for measuring unemployment.
384. M.J. VAN DER LEIJ, The economics of networks: Theory and empirics.

385. R. VAN DER NOLL, Essays on internet and information economics.
386. V. PANCHENKO; Nonparametric methods in economics and finance: dependence, causality and prediction.
387. C.A.S.P. SA, Higher education choice in The Netherlands: The economics of where to go.
388. J. DELFGAAUW, Wonderful and woeful work: Incentives, selection, turnover, and workers' motivation.
389. G. DEBREZION, Railway impacts on real estate prices.
390. A.V. HARDIYANTO, Time series studies on Indonesian rupiah/USD rate 1995 - 2005.
391. M.I.S.H. MUNANDAR, Essays on economic integration.
392. K.G. BERDEN, On technology, uncertainty and economic growth.
393. G. VAN DE KUILEN, The economic measurement of psychological risk attitudes.
394. E.A. MOOI, Inter-organizational cooperation, conflict, and change.
395. A. LLENA NOZAL, On the dynamics of health, work and socioeconomic status.
396. P.D.E. DINDO, Bounded rationality and heterogeneity in economic dynamic models.
397. D.F. SCHRAGER, Essays on asset liability modeling.
398. R. HUANG, Three essays on the effects of banking regulations.
399. C.M. VAN MOURIK, Globalisation and the role of financial accounting information in Japan.
400. S.M.S.N. MAXIMIANO, Essays in organizational economics.
401. W. JANSSENS, Social capital and cooperation: An impact evaluation of a women's empowerment programme in rural India.
402. J. VAN DER SLUIS, Successful entrepreneurship and human capital.
403. S. DOMINGUEZ MARTINEZ, Decision making with asymmetric information.
404. H. SUNARTO, Understanding the role of bank relationships, relationship marketing, and organizational learning in the performance of people's credit bank.
405. M.A. DOS REIS PORTELA, Four essays on education, growth and labour economics.
406. S.S. FICCO, Essays on imperfect information-processing in economics.
407. P.J.P.M. VERSIJP, Advances in the use of stochastic dominance in asset pricing.

408. M.R. WILDENBEEST, Consumer search and oligopolistic pricing: A theoretical and empirical inquiry.

409. E. GUSTAFSSON-WRIGHT, Baring the threads: Social capital, vulnerability and the well-being of children in Guatemala.

410. S. YERGOU-WORKU, Marriage markets and fertility in South Africa with comparisons to Britain and Sweden.

411. J.F. SLIJKERMAN, Financial stability in the EU.

412. W.A. VAN DEN BERG, Private equity acquisitions.

413. Y. CHENG, Selected topics on nonparametric conditional quantiles and risk theory.

414. M. DE POOTER, Modeling and forecasting stock return volatility and the term structure of interest rates.

415. F. RAVAZZOLO, Forecasting financial time series using model averaging.

416. M.J.E. KABKI, Transnationalism, local development and social security: the functioning of support networks in rural Ghana.

417. M. POPLAWSKI RIBEIRO, Fiscal policy under rules and restrictions.

418. S.W. BISSESSUR, Earnings, quality and earnings management: the role of accounting accruals.

419. L. RATNOVSKI, A Random Walk Down the Lombard Street: Essays on Banking.

420. R.P. NICOLAI, Maintenance models for systems subject to measurable deterioration.

421. R.K. Andadari, Local clusters in global value chains, a case study of wood furniture clusters in Central Java (Indonesia).

SIKS Disseration Series

1998

1998-1 Johan van den Akker (CWI)
DEGAS - An Active, Temporal Database of Autonomous Objects

1998-2 Floris Wiesman (UM)
Information Retrieval by Graphically Browsing Meta-Information

1998-3 Ans Steuten (TUD)
A Contribution to the Linguistic Analysis of Business Conversations within the Language/Action Perspective

1998-4 Dennis Breuker (UM)
Memory versus Search in Games

1998-5 E.W.Oskamp (RUL)
Computerondersteuning bij Straftoemeting

1999

1999-1 Mark Sloof (VU)
Physiology of Quality Change Modelling; Automated modelling of Quality Change of Agricultural Products

1999-2 Rob Potharst (EUR)
Classification using decision trees and neural nets

1999-3 Don Beal (UM)
The Nature of Minimax Search

1999-4 Jacques Penders (UM)
The practical Art of Moving Physical Objects

1999-5 Aldo de Moor (KUB)
Empowering Communities: A Method for the Legitimate User-Driven Specification of Network Information Systems

1999-6 Niek J.E. Wijngaards (VU)
Re-design of compositional systems

1999-7 David Spelt (UT)
Verification support for object database design

1999-8 Jacques H.J. Lenting (UM)
Informed Gambling: Conception and Analysis of a Multi-Agent Mechanism for Discrete Reallocation.

2000

2000-1 Frank Niessink (VU)
Perspectives on Improving Software Maintenance

2000-2 Koen Holtman (TUE)
Prototyping of CMS Storage Management

2000-3 Carolien M.T. Metselaar (UVA)
Sociaal-organisatorische gevolgen van kennistechnologie; een procesbenadering en actorperspectief

2000-4 Geert de Haan (VU)
ETAG, A Formal Model of Competence Knowledge for User Interface Design

2000-5 Ruud van der Pol (UM)
Knowledge-based Query Formulation in Information Retrieval

2000-6 Rogier van Eijk (UU)
Programming Languages for Agent Communication

2000-7 Niels Peek (UU)
Decision-theoretic Planning of Clinical Patient Management

2000-8 Veerle Coup (EUR)
Sensitivity Analyis of Decision-Theoretic Networks

2000-9 Florian Waas (CWI)
Principles of Probabilistic Query Optimization

2000-10 Niels Nes (CWI)
Image Database Management System Design Considerations, Algorithms and Architecture

2000-11 Jonas Karlsson (CWI)
Scalable Distributed Data Structures for Database Management

2001

2001-1 Silja Renooij (UU)
Qualitative Approaches to Quantifying Probabilistic Networks

2001-2 Koen Hindriks (UU)
Agent Programming Languages: Programming with Mental Models

2001-3 Maarten van Someren (UvA)
Learning as problem solving

2001-4 Evgueni Smirnov (UM)
Conjunctive and Disjunctive Version Spaces with Instance-Based Boundary Sets

2001-5 Jacco van Ossenbruggen (VU)
Processing Structured Hypermedia: A Matter of Style

2001-6 Martijn van Welie (VU)
Task-based User Interface Design

2001-7 Bastiaan Schonhage (VU)
Diva: Architectural Perspectives on Information Visualization

2001-8 Pascal van Eck (VU)
A Compositional Semantic Structure for Multi-Agent Systems Dynamics.

2001-9 Pieter Jan 't Hoen (RUL)
Towards Distributed Development of Large Object-Oriented Models, Views of Packages as Classes

2001-10 Maarten Sierhuis (UvA)
Modeling and Simulating Work Practice BRAHMS: a multiagent modeling and simulation language for work practice analysis and design

2001-11 Tom M. van Engers (VUA)
Knowledge Management: The Role of Mental Models in Business Systems Design

2002

2002-01 Nico Lassing (VU)
Architecture-Level Modifiability Analysis

2002-02 Roelof van Zwol (UT)
Modelling and searching web-based document collections

2002-03 Henk Ernst Blok (UT)
Database Optimization Aspects for Information Retrieval

2002-04 Juan Roberto Castelo Valdueza (UU)
The Discrete Acyclic Digraph Markov Model in Data Mining

2002-05 Radu Serban (VU)
The Private Cyberspace Modeling Electronic Environments inhabited by Privacy-concerned Agents

2002-06 Laurens Mommers (UL)
Applied legal epistemology; Building a knowledge-based ontology of the legal domain

2002-07 Peter Boncz (CWI)
Monet: A Next-Generation DBMS Kernel For Query-Intensive Applications

2002-08 Jaap Gordijn (VU)
Value Based Requirements Engineering: Exploring Innovative E-Commerce Ideas

2002-09 Willem-Jan van den Heuvel(KUB)
Integrating Modern Business Applications with Objectified Legacy Systems

2002-10 Brian Sheppard (UM)
Towards Perfect Play of Scrabble

2002-11 Wouter C.A. Wijngaards (VU)
Agent Based Modelling of Dynamics: Biological and Organisational Applications

2002-12 Albrecht Schmidt (Uva)
Processing XML in Database Systems

2002-13 Hongjing Wu (TUE)
A Reference Architecture for Adaptive Hypermedia Applications

2002-14 Wieke de Vries (UU)
Agent Interaction: Abstract Approaches to Modelling, Programming and Verifying Multi-Agent Systems

2002-15 Rik Eshuis (UT)
Semantics and Verification of UML Activity Diagrams for Workflow Modelling

2002-16 Pieter van Langen (VU)
The Anatomy of Design: Foundations, Models and Applications

2002-17 Stefan Manegold (UVA)
Understanding, Modeling, and Improving Main-Memory Database Performance

2003

2003-01 Heiner Stuckenschmidt (VU)
Ontology-Based Information Sharing in Weakly Structured Environments

2003-02 Jan Broersen (VU)
Modal Action Logics for Reasoning About Reactive Systems

2003-03 Martijn Schuemie (TUD)
Human-Computer Interaction and Presence in Virtual Reality Exposure Therapy

2003-04 Milan Petkovic (UT)
Content-Based Video Retrieval Supported by Database Technology

2003-05 Jos Lehmann (UVA)
Causation in Artificial Intelligence and Law - A modelling approach

2003-06 Boris van Schooten (UT)
Development and specification of virtual environments

2003-07 Machiel Jansen (UvA)
Formal Explorations of Knowledge Intensive Tasks

2003-08 Yongping Ran (UM)
Repair Based Scheduling

2003-09 Rens Kortmann (UM)
The resolution of visually guided behaviour

2003-10 Andreas Lincke (UvT)
Electronic Business Negotiation: Some experimental studies on the interaction between medium, innovation context and culture

2003-11 Simon Keizer (UT)
Reasoning under Uncertainty in Natural Language Dialogue using Bayesian Networks

2003-12 Roeland Ordelman (UT)
Dutch speech recognition in multimedia information retrieval

2003-13 Jeroen Donkers (UM)
Nosce Hostem - Searching with Opponent Models

2003-14 Stijn Hoppenbrouwers (KUN)
Freezing Language: Conceptualisation Processes across ICT-Supported Organisations

2003-15 Mathijs de Weerdt (TUD)
Plan Merging in Multi-Agent Systems

2003-16 Menzo Windhouwer (CWI)
Feature Grammar Systems - Incremental Maintenance of Indexes to Digital Media Warehouses

2003-17 David Jansen (UT)
Extensions of Statecharts with Probability, Time, and Stochastic Timing

2003-18 Levente Kocsis (UM)
Learning Search Decisions

2004

2004-01 Virginia Dignum (UU)
A Model for Organizational Interaction: Based on Agents, Founded in Logic

2004-02 Lai Xu (UvT)
Monitoring Multi-party Contracts for E-business

2004-03 Perry Groot (VU)
A Theoretical and Empirical Analysis of Approximation in Symbolic Problem Solving

2004-04 Chris van Aart (UVA)
Organizational Principles for Multi-Agent Architectures

2004-05 Viara Popova (EUR)
Knowledge discovery and monotonicity

2004-06 Bart-Jan Hommes (TUD)
The Evaluation of Business Process Modeling Techniques

2004-07 Elise Boltjes (UM)
Voorbeeldig onderwijs; voorbeeldgestuurd onderwijs, een opstap naar abstract denken, vooral voor meisjes

2004-08 Joop Verbeek(UM)
Politie en de Nieuwe Internationale Informatiemarkt, Grensregionale politile gegevensuitwisseling en digitale expertise

2004-09 Martin Caminada (VU)
For the Sake of the Argument; explorations into argument-based reasoning

2004-10 Suzanne Kabel (UVA)
Knowledge-rich indexing of learning-objects

2004-11 Michel Klein (VU)
Change Management for Distributed Ontologies

2004-12 The Duy Bui (UT)
Creating emotions and facial expressions for embodied agents

2004-13 Wojciech Jamroga (UT)
Using Multiple Models of Reality: On Agents who Know how to Play

2004-14 Paul Harrenstein (UU)
Logic in Conflict. Logical Explorations in Strategic Equilibrium

2004-15 Arno Knobbe (UU)
Multi-Relational Data Mining

2004-16 Federico Divina (VU)
Hybrid Genetic Relational Search for Inductive Learning

2004-17 Mark Winands (UM)
Informed Search in Complex Games

2004-18 Vania Bessa Machado (UvA)
Supporting the Construction of Qualitative Knowledge Models

2004-19 Thijs Westerveld (UT)
Using generative probabilistic models for multimedia retrieval

2004-20 Madelon Evers (Nyenrode)
Learning from Design: facilitating multidisciplinary design teams

2005

2005-01 Floor Verdenius (UVA)
Methodological Aspects of Designing Induction-Based Applications

2005-02 Erik van der Werf (UM)
AI techniques for the game of Go

2005-03 Franc Grootjen (RUN)
A Pragmatic Approach to the Conceptualisation of Language

2005-04 Nirvana Meratnia (UT)
Towards Database Support for Moving Object data

2005-05 Gabriel Infante-Lopez (UVA)
Two-Level Probabilistic Grammars for Natural Language Parsing

2005-06 Pieter Spronck (UM)
Adaptive Game AI

2005-07 Flavius Frasincar (TUE)
Hypermedia Presentation Generation for Semantic Web Information Systems

2005-08 Richard Vdovjak (TUE)
A Model-driven Approach for Building Distributed Ontology-based Web Applications

2005-09 Jeen Broekstra (VU)
Storage, Querying and Inferencing for Semantic Web Languages

2005-10 Anders Bouwer (UVA)
Explaining Behaviour: Using Qualitative Simulation in Interactive Learning Environments

2005-11 Elth Ogston (VU)
Agent Based Matchmaking and Clustering - A Decentralized Approach to Search

2005-12 Csaba Boer (EUR)
Distributed Simulation in Industry

2005-13 Fred Hamburg (UL)
Een Computermodel voor het Ondersteunen van Euthanasiebeslissingen

2005-14 Borys Omelayenko (VU)
Web-Service configuration on the Semantic Web; Exploring how semantics meets pragmatics

2005-15 Tibor Bosse (VU)
Analysis of the Dynamics of Cognitive Processes

2005-16 Joris Graaumans (UU)
Usability of XML Query Languages

2005-17 Boris Shishkov (TUD)
Software Specification Based on Re-usable Business Components

2005-18 Danielle Sent (UU)
Test-selection strategies for probabilistic networks

2005-19 Michel van Dartel (UM)
Situated Representation

2005-20 Cristina Coteanu (UL)
Cyber Consumer Law, State of the Art and Perspectives

2005-21 Wijnand Derks (UT)
Improving Concurrency and Recovery in Database Systems by Exploiting Application Semantics

2006

2006-01 Samuil Angelov (TUE)
Foundations of B2B Electronic Contracting

2006-02 Cristina Chisalita (VU)
Contextual issues in the design and use of information technology in organizations

2006-03 Noor Christoph (UVA)
The role of metacognitive skills in learning to solve problems

2006-04 Marta Sabou (VU)
Building Web Service Ontologie

2006-05 Cees Pierik (UU)
Validation Techniques for Object-Oriented Proof Outlines

2006-06 Ziv Baida (VU)
Software-aided Service Bundling - Intelligent Methods & Tools for Graphical Service Modeling

2006-07 Marko Smiljanic (UT)
XML schema matching – balancing efficiency and effectiveness by means of clustering

2006-08 Eelco Herder (UT)
Forward, Back and Home Again - Analyzing User Behavior on the Web

2006-09 Mohamed Wahdan (UM)
Automatic Formulation of the Auditor's Opinion

2006-10 Ronny Siebes (VU)
Semantic Routing in Peer-to-Peer Systems

2006-11 Joeri van Ruth (UT)
Flattening Queries over Nested Data Types

2006-12 Bert Bongers (VU)
Interactivation - Towards an e-cology of people, our technological environment, and the arts

2006-13 Henk-Jan Lebbink (UU)
Dialogue and Decision Games for Information Exchanging Agents

2006-14 Johan Hoorn (VU)
Software Requirements: Update, Upgrade, Redesign - towards a Theory of Requirements Change

2006-15 Rainer Malik (UU)
CONAN: Text Mining in the Biomedical Domain

2006-16 Carsten Riggelsen (UU)
Approximation Methods for Efficient Learning of Bayesian Networks

2006-17 Stacey Nagata (UU)
User Assistance for Multitasking with Interruptions on a Mobile Device

2006-18 Valentin Zhizhkun (UVA)
Graph transformation for Natural Language Processing

2006-19 Birna van Riemsdijk (UU)
Cognitive Agent Programming: A Semantic Approach

2006-20 Marina Velikova (UvT)
Monotone models for prediction in data mining

2006-21 Bas van Gils (RUN)
Aptness on the Web

2006-22 Paul de Vrieze (RUN)
Fundaments of Adaptive Personalisation

2006-23 Ion Juvina (UU)
Development of Cognitive Model for Navigating on the Web

2006-24 Laura Hollink (VU)
Semantic Annotation for Retrieval of Visual Resources

2006-25 Madalina Drugan (UU)
Conditional log-likelihood MDL and Evolutionary MCMC

2006-26 Vojkan Mihajlovic (UT)
Score Region Algebra: A Flexible Framework for Structured Information Retrieval

2006-27 Stefano Bocconi (CWI)
Vox Populi: generating video documentaries from semantically annotated media repositories

2006-28 Borkur Sigurbjornsson (UVA)
Focused Information Access using XML Element Retrieval

2007

2007-01 Kees Leune (UvT)
Access Control and Service-Oriented Architectures

2007-02 Wouter Teepe (RUG)
Reconciling Information Exchange and Confidentiality: A Formal Approach

2007-03 Peter Mika (VU)
Social Networks and the Semantic Web

2007-04 Jurriaan van Diggelen (UU)
Achieving Semantic Interoperability in Multi-agent Systems: a dialogue-based approach

2007-05 Bart Schermer (UL)
Software Agents, Surveillance, and the Right to Privacy: a Legislative Framework for Agent-enabled Surveillance

2007-06 Gilad Mishne (UVA)
Applied Text Analytics for Blogs

2007-07 Natasa Jovanovic' (UT)
To Whom It May Concern - Addressee Identification in Face-to-Face Meetings

2007-08 Mark Hoogendoorn (VU)
Modeling of Change in Multi-Agent Organizations

2007-09 David Mobach (VU)
Agent-Based Mediated Service Negotiation

2007-10 Huib Aldewereld (UU)
Autonomy vs. Conformity: an Institutional Perspective on Norms and Protocols

2007-11 Natalia Stash (TUE)
Incorporating Cognitive/Learning Styles in a General-Purpose Adaptive Hypermedia System

2007-12 Marcel van Gerven (RUN)
Bayesian Networks for Clinical Decision Support: A Rational Approach to Dynamic Decision-Making under Uncertainty

2007-13 Rutger Rienks (UT)
Meetings in Smart Environments; Implications of Progressing Technology

2007-14 Niek Bergboer (UM)
Context-Based Image Analysis

2007-15 Joyca Lacroix (UM)
NIM: a Situated Computational Memory Model

2007-16 Davide Grossi (UU)
Designing Invisible Handcuffs. Formal investigations in Institutions and Organizations for Multi-agent Systems

2007-17 Theodore Charitos (UU)
Reasoning with Dynamic Networks in Practice

2007-18 Bart Orriens (UvT)
On the development an management of adaptive business collaborations

2007-19 David Levy (UM)
Intimate relationships with artificial partners

2007-20 Slinger Jansen (UU)
Customer Configuration Updating in a Software Supply Network

2007-21 Karianne Vermaas (UU)
Fast diffusion and broadening use: A research on residential adoption and usage of broadband internet in the Netherlands between 2001 and 2005

2007-22 Zlatko Zlatev (UT)
Goal-oriented design of value and process models from patterns

2007-23 Peter Barna (TUE)
Specification of Application Logic in Web Information Systems

2007-24 Georgina Ramirez Camps (CWI)
Structural Features in XML Retrieval

2007-25 Joost Schalken (VU)
Empirical Investigations in Software Process Improvement

2008

2008-01 Katalin Boer-Sorbn (EUR)
Agent-Based Simulation of Financial Markets: A modular, continuous-time approach

2008-02 Alexei Sharpanskykh (VU)
On Computer-Aided Methods for Modeling and Analysis of Organizations

2008-03 Vera Hollink (UVA)
Optimizing hierarchical menus: a usage-based approach

2008-04 Ander de Keijzer (UT)
Management of Uncertain Data - towards unattended integration

2008-05 Bela Mutschler (UT)
Modeling and simulating causal dependencies on process-aware information systems from a cost perspective

2008-06 Arjen Hommersom (RUN)
On the Application of Formal Methods to Clinical Guidelines, an Artificial Intelligence Perspective

2008-07 Peter van Rosmalen (OU)
Supporting the tutor in the design and support of adaptive e-learning

2008-08 Janneke Bolt (UU)
Bayesian Networks: Aspects of Approximate Inference

2008-09 Christof van Nimwegen (UU)
The paradox of the guided user: assistance can be counter-effective

2008-10 Wauter Bosma (UT)
Discourse oriented summarization