Information Systems Actability

Understanding Information Technology as a Tool for Business Action and Communication

Pär J. Ågerfalk

Department of Computer and Information Science
Linköping University
SE-581 83 Linköping, Sweden

Linköping 2003
‘My words fly up, my thoughts remain below: Words without thoughts never to heaven go.’

William Shakespeare’s Hamlet, Act III, Sc. 3
Abstract

This dissertation is devoted to a perspective from which IT-based information systems are conceived as information technological artefacts intended for business action and communication. The perspective has been made concrete through the concept of information systems actability, which is the main concept under scrutiny. The research contributions of the dissertation exist at various levels of abstraction. First of all, the dissertation contributes to the understanding of information systems from a social action perspective. Based on identified weaknesses in contemporary conceptualizations of information systems, the concept of information systems actability is developed. This contribution consists of a reconciliation of various views on information system usage quality with its roots in a linguistic tradition including elements from organizational semiotics and the language/action perspective. At a more concrete level, this understanding, and the concept as such, have consequences for the development and evaluation of information systems. Such consequences have been the foundation for a proposed information systems design method, which thus constitutes a further contribution. Another contribution is the outlining of an information systems evaluation approach based on the concept of information systems actability. As a further means of obtaining empirical experience of working with this concept, a descriptive analytic framework has been developed, which constitutes yet another contribution. These three operationalizations, the design method, the evaluation method and the analytic framework, have been developed and empirically grounded through a qualitative case study approach involving four cases of information systems requirements specification, four evaluations of existing information systems, and two cases of description and characterization of information systems related phenomena from the perspective of actability. The latter two cases imply two further contributions at an even more concrete level, constituted by characterizations of Internet-based information systems and the local electronic marketplace, seen in the light of information systems actability.
Foreword

Information systems development is a discipline within the Faculty of Arts and Sciences at Linköping University. Information systems development is a discipline studying human work with developing and changing computer-based information systems in organisational settings. It includes theories, strategies, models, methods, co-working principles and tools concerning information systems development. Different development/change situations can be studied as planning, analysis, specification, design, implementation, deployment, evaluation, maintenance and redesign of information systems and its interplay with other forms of business development. The discipline also includes the study of prerequisites for and results from information systems development, as e.g. studies of usage and consequences of information systems.

This work, Information Systems Actability: Understanding Information Technology as a Tool for Business Action and Communication, is written by Pär J. Ågerfalk at Örebro University. He is also a member of research group VITS. He presents this work as his Ph D dissertation in information systems development, Department of Computer and Information Science, Linköping University.

Linköping March 2003

Göran Goldkuhl
Professor
Doctoral Dissertations in Information Systems Development


Licentiate theses in Information Systems Development


23. Stefan Holgersson (2001) IT-system och filtrering av verksamhetskunskap – kvalitetsproblem vid analyser och beslutsfattande som bygger på uppgifter hämtade från polisens IT-system

24. Per Oscarson (2001) Informationssäkerhet i verksamheter – begrepp och modeller som stöd för förståelse av informationssäkerhet och dess hantering i verksamheter


Acknowledgements

Writing a doctoral dissertation can be a frustrating and lonely endeavour. Long hours at work, remote comments from unappreciative supervisors, and the endless darkness of the Nordic winter outside the frosty window of that scabby cubbyhole they insist on calling an office. Or, it can be an inspiring and intellectually stimulating joint venture. ‘To boldly go where no man has gone before,’ bringing your best friends along the way! I have experienced both. The truth is, had it not been for all the encouraging people around me, I would have given up long ago. Sucked into the Nordic night and never forgiving myself for turning down that job offer long before the IT crisis blew the guts out of every risk capitalist in sight. In an attempt to make up at least a bit to all these people, I will here try to mention some of you, even though I know that such a list can never be complete and that I will most likely spend the rest of my days banging my head against a brick wall in vain for having missed some of the most important people out. Here we go:

Malin, fiancée, best friend, and spiritual guide. I have not, cannot, and will never understand how you manage to put up with me, and my work addiction. Thank you for always being there!

Göran Goldkuhl, supervisor, and Stefan Cronholm, co-supervisor. Without your competence, encouragement and endless support, this dissertation would never have been written.

Jan Gulliksen, co-supervisor for a short period. Thanks for your critical comments!

Anders Avdic, local supervisor at Örebro University. Thanks for making me go to Linköping and talk to Göran in the first place – this is actually due to you!

Johan Petersson, Emma Eliason, Jonas Sjöström, Fredrik Karlsson, Anders Hjalmarsson, Owen Eriksson, Göran Goldkuhl, and Stefan Cronholm, co-authors of the papers upon which this dissertation is based. Let’s face it, we have actually written this dissertation together!

Kenneth Åhlgren, Per Oscarsson, Kai Wistrand, Gregor Börjesson, Martin Elmberg, and Kjell Möller, co-authors of other papers produced along the way. Maybe I should have written another dissertation? Thanks also to Pär Fahlén, we did never write anything, but at least we had fun!

All the people I have had the opportunity to work with in different empirical settings. I would specifically like to mention Torsten Gille, Lasse Gustafsson, Fredrik Gustafsson, Stefan Stråle, Magnus Fröberg, Odd Ingvarsson, Carina Angelsten, and Ulla Arsenius. Thanks for making the work pleasant and valuable.

All my colleagues at Örebro University, Linköping University and in the Research Network VITS who have suffered through the many seminar presentations and given valuable feedback throughout the entire research process. Specifically I owe thanks to all of you who gave me important feedback in the final phase of this work: Karin Hed-

All the anonymous reviewers of the papers upon which this dissertation is based, and all the active participants at the conferences where I have presented my ideas. Your questions, comments and suggestions have been most valuable.

Lillemor Wallgren and Britt-Inger Karlsson, graduate studies administrators. How do you manage always to do everything with a smile?

Simon Lim, martial arts instructor, Chris Bunkell, acupuncturist, and Ann-Marie Earey and Sassa Lee, Ashtanga Yoga teachers. Your guidance and treatment have definitely helped to keep my stress level from hitting the ceiling.

Finally, Tom Lacy for proofreading the final manuscript.

This work has been financially supported by The Knowledge Foundation (The KK-Foundation) through the program for the promotion of research in IT at new universities and university colleges in Sweden (IT-Lyftet), Örebro University, The Swedish National Board for Industrial and Technical Development (NUTEK) through the MCC-project (Management system for change-over efficiency in complex business and production processes), and The Swedish Agency for Innovation Systems (VINNOVA), formerly The Swedish Council for Work Life Research (RALF), through the project Actable Information Systems: Through Evaluation and Redesign.

May the illocutionary force be with you!

Örebro, Sweden, March 2003

Pär J. Ågerfalk
# Content

## PART I: PRELIMINARIES

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Introduction</td>
<td>3</td>
</tr>
<tr>
<td>1.1 Information Systems Usage Quality</td>
<td>3</td>
</tr>
<tr>
<td>1.2 Contextualization of Information Systems</td>
<td>5</td>
</tr>
<tr>
<td>1.2.1 Usability for Usage Quality</td>
<td>5</td>
</tr>
<tr>
<td>1.2.2 The Requirements Gap</td>
<td>7</td>
</tr>
<tr>
<td>1.2.3 The Cognitive versus the Organizational Approach</td>
<td>8</td>
</tr>
<tr>
<td>1.2.4 Information System Users</td>
<td>10</td>
</tr>
<tr>
<td>1.3 Pragmatization of Information Systems</td>
<td>10</td>
</tr>
<tr>
<td>1.3.1 The Usage Fallacy</td>
<td>10</td>
</tr>
<tr>
<td>1.3.2 The Instrumental Fallacy</td>
<td>11</td>
</tr>
<tr>
<td>1.3.3 The Descriptive Fallacy</td>
<td>12</td>
</tr>
<tr>
<td>1.4 Aim of Research</td>
<td>12</td>
</tr>
<tr>
<td>1.4.1 Detailing the Research Aim</td>
<td>12</td>
</tr>
<tr>
<td>1.4.2 A Summary of Research Contributions</td>
<td>14</td>
</tr>
<tr>
<td>1.5 Demarcations</td>
<td>14</td>
</tr>
<tr>
<td>1.6 A Note on Related Research</td>
<td>15</td>
</tr>
<tr>
<td>1.7 Dissertation Structure and Research Setting</td>
<td>15</td>
</tr>
<tr>
<td>1.7.1 Introducing Part I of the Dissertation</td>
<td>17</td>
</tr>
<tr>
<td>1.7.2 Introducing Part II of the Dissertation</td>
<td>17</td>
</tr>
<tr>
<td>1.7.3 Introducing Part III of the Dissertation</td>
<td>18</td>
</tr>
<tr>
<td>1.7.4 Introducing Part IV of the Dissertation</td>
<td>18</td>
</tr>
<tr>
<td>1.7.5 Introducing Part V of the Dissertation</td>
<td>18</td>
</tr>
<tr>
<td>1.7.6 Introducing Part VI of the Dissertation</td>
<td>19</td>
</tr>
<tr>
<td>1.8 Intended Audience and Reading Instructions</td>
<td>19</td>
</tr>
<tr>
<td>2 Field of Study</td>
<td>21</td>
</tr>
<tr>
<td>2.1 Scientific Abode</td>
<td>21</td>
</tr>
<tr>
<td>2.2 Information Systems Theory and This Work</td>
<td>22</td>
</tr>
<tr>
<td>2.3 The Concept of Method in Systems Development</td>
<td>24</td>
</tr>
<tr>
<td>2.3.1 Method versus Methodology</td>
<td>25</td>
</tr>
<tr>
<td>2.3.2 Other Method-Related Concepts</td>
<td>26</td>
</tr>
<tr>
<td>2.3.3 Relating the Concepts</td>
<td>29</td>
</tr>
<tr>
<td>2.3.4 Existence Levels of Methods</td>
<td>31</td>
</tr>
<tr>
<td>2.3.5 Method development</td>
<td>32</td>
</tr>
<tr>
<td>2.4 Self Criticism</td>
<td>33</td>
</tr>
<tr>
<td>2.4.1 Yet Another ‘ability’?</td>
<td>33</td>
</tr>
<tr>
<td>2.4.2 Yet Another Method?</td>
<td>33</td>
</tr>
<tr>
<td>2.4.3 Why a Method-Based Approach?</td>
<td>34</td>
</tr>
<tr>
<td>3 Research Design</td>
<td>36</td>
</tr>
<tr>
<td>3.1 A Setting for Interpretative in-Context Research</td>
<td>36</td>
</tr>
<tr>
<td>3.2 Conceptualization and Application of Knowledge</td>
<td>38</td>
</tr>
<tr>
<td>3.2.1 Ontological Foundation</td>
<td>38</td>
</tr>
<tr>
<td>3.2.2 Principles for Grounding of Knowledge</td>
<td>40</td>
</tr>
<tr>
<td>3.2.3 Concepts, Operationalizations, Applications and Consequences</td>
<td>41</td>
</tr>
<tr>
<td>3.3 Research Strategy</td>
<td>42</td>
</tr>
<tr>
<td>3.4 The Grounding of Actability</td>
<td>46</td>
</tr>
<tr>
<td>3.5 The Research Process</td>
<td>47</td>
</tr>
</tbody>
</table>
3.6 A Note on Relevance, Reliability and Validity ........................................ 48
3.7 The Validity of Actability ........................................................................ 49

PART II: THE CONCEPT OF INFORMATION SYSTEMS ACTABILITY .......... 51

4 A Pragmatic View on Information Systems .............................................. 53
4.1 Chapter Outline .................................................................................... 53
4.2 Setting the Scene .................................................................................. 53
4.3 Information Systems and Pragmatics ..................................................... 54
4.4 Information Systems: Action Artefacts in Organizations ...................... 57
  4.4.1 Human Action .................................................................................. 57
  4.4.2 Language Acts and Other Communication Acts .............................. 61
  4.4.3 Organizational Action ..................................................................... 61
  4.4.4 The Nature of Artefacts ................................................................. 62
  4.4.5 IT-Based Information Systems ........................................................ 62
4.5 Introducing Information Systems Actability ........................................... 64
  4.5.1 Information System Use-Situations .................................................. 65
  4.5.2 Elementary Interactions .................................................................. 68
  4.5.3 Interactive Use-Situations, Tasks and Use Cases ............................. 74
  4.5.4 Action Memory and Databases ....................................................... 74
  4.5.5 Actability Features of Information Systems ..................................... 75
4.6 Implications for Information Systems Work .......................................... 77
4.7 Summarizing the Presented View ......................................................... 77
4.8 Criticisms of Speech Act Theory ............................................................ 79

5 Messages in Information Systems ............................................................. 82
5.1 Introduction ......................................................................................... 82
5.2 From Description to Language/Action .................................................. 83
5.3 Actions and Messages in Context ......................................................... 86
   5.3.1 The Concept of a Speech Act ......................................................... 86
   5.3.2 The Multi-Functionality of Speech Acts ........................................ 87
5.4 The Action-Elementary Message .......................................................... 88
   5.4.1 A preliminary View of the Action-Elementary Message .............. 89
   5.4.2 An Elaborated View of the Action-Elementary Message ............ 92
   5.4.3 A Rejected View of the Action-Elementary Message ................. 95
5.5 Introducing the IT Artefact ................................................................. 95
   5.5.1 Participants in the Conversation .................................................... 96
   5.5.2 Input Messages and Output Messages ....................................... 96
5.6 Messages in Information Systems – A Summary ................................... 99

6 Pragmatization of Conceptual Modelling ............................................... 101
6.1 Introduction ......................................................................................... 101
6.2 Beyond the Descriptive Perspective .................................................... 102
6.3 Information Systems and Speech Acts ................................................. 103
6.4 Conversation for Action ...................................................................... 105
6.5 Action-Oriented Conceptual Modelling ............................................... 106
   6.5.1 Dynamic Modelling ................................................................. 107
   6.5.2 Static Modelling ......................................................................... 110
6.6 Pragmatization of Conceptual Modelling – Conclusions ..................... 113

7 Usability in Social Action ...................................................................... 115
7.1 Introduction to Usability in Social Action ........................................... 115
7.2 The Contemporary Instrumental View on Usability ............................ 116
10 Actability Design: Evolution and Empirical Lessons Learned

10.1 Evolution of VIBA

10.1.1 From VIBA’93 to VIBA’02

10.1.2 Process Modelling versus VIBA’93

10.1.3 Action Modelling versus VIBA’93

10.1.4 Interaction Modelling versus VIBA’93

10.1.5 VIBA’98 versus VIBA’02

10.1.6 VIBA’99 versus VIBA’02

10.1.7 VIBA’00 versus VIBA’02

10.2 From Actability to VIBA

10.2.1 Social Action as the Starting Point

10.2.2 Speech Acts and Interaction Modelling

10.3 Empirical VIBA Lessons Learned

10.3.1 Process

10.3.2 Product

10.3.3 Concluding Remarks

10.4 Overall Reflections on the Research on VIBA

11 Research on Actability Evaluation

11.1 Actability Evaluation Background

11.2 Actability Evaluation Assumptions and Design Goals

11.3 Actability Evaluation Research Approach

11.4 About the Actability Evaluation Applications

11.4.1 Application 5: The Internet Bank

11.4.2 Application 6: The Booking System

11.4.3 Application 7: The Intranet

11.4.4 Application 8: The Scheduling System

11.5 Actability Evaluation Data Collection

11.5.1 Selection of Applications

11.5.2 Records

11.6 Actability Evaluation Data Analysis

12 Evaluation of Information Systems in Context

12.1 Introduction

12.2 Evaluation Process Model

12.2.1 Ideal Typical Analysis

12.2.2 Situational Analysis

12.3 Analytical Tools for Evaluation

12.3.1 A Set of Actability Principles

12.3.2 The D.E.U.P.S. Model

12.3.3 A Layered Model of Action

12.4 Documenting the Evaluation

12.5 A Summary of the Proposed Evaluation Approach

13 Actability Evaluation: Evolution and Empirical Lessons Learned

13.1 Evolution of Actability Evaluation

13.2 From Actability to Actability Evaluation

13.2.1 Grounding of the Actability Principles

13.2.2 The D.E.U.P.S. Model and Actability

13.2.3 Actability and the Layered Model of Action
PART V: ACTABILITY STUDIES ................................................................. 243

14 Study I: The Case of Internet-Based Systems ...................................... 245
  14.1 Chapter Outline .............................................................................. 245
  14.2 Background of Study I ................................................................. 245
  14.3 Actability Study I Research Method ................................................ 247
  14.4 An Analytic Framework Based on Actability ................................. 248
    14.4.1 The Semiotic Framework .......................................................... 249
    14.4.2 Performance of Action Revisited ............................................... 250
    14.4.3 Information Systems: Software Artefacts for Human Action ...... 252
    14.4.4 Information System Users as Business Actors .......................... 252
    14.4.5 The Business Context ............................................................... 252
    14.4.6 Summarizing the Analytic Framework ...................................... 253
  14.5 Internet-Based Systems According to Actability ............................... 254
    14.5.1 The Actors ............................................................................... 255
    14.5.2 The Artefact ............................................................................ 256
  14.6 Concluding Actability Study I ....................................................... 261

15 Study II: Multi-Channel Interplay in Practice .................................... 263
  15.1 Introduction to Study II ................................................................. 263
  15.2 Actability Study II Research Method .............................................. 264
    15.2.1 An Elaborated Analytic Framework Based on Actability .......... 265
    15.2.2 Description of Marketplaces Studied ....................................... 266
  15.3 The Multi-Channel Interplay of Local Commerce ............................ 266
    15.3.1 Electronic Marketplaces for Local Commerce ........................... 267
    15.3.2 Practising Marketplace ............................................................ 268
  15.4 Actable Systems for Interplay .......................................................... 269
    15.4.1 Action through Multiple Channels ........................................... 269
    15.4.2 Utilizing the System ................................................................. 270
    15.4.3 Multi-Channel Actors ............................................................... 271
  15.5 Concluding Actability Study II ....................................................... 271

PART VI: CONCLUSION ............................................................................. 273

16 Summing Up and Planning for the Future ......................................... 275
  16.1 Actability as a Perspective on Information Systems ......................... 275
  16.2 Revisiting the Point of Departure .................................................... 276
    16.2.1 Information Systems Usage Quality Revisited ......................... 277
    16.2.2 Actability and Contextualization of Information Systems .......... 278
    16.2.3 Actability and Pragmatization of Information Systems .............. 279
  16.3 Does Actability Reinforce or Reinvent? ......................................... 280
    16.3.1 Actability and Usability ............................................................ 280
    16.3.2 Actability and Communication Modelling .................................. 282
    16.3.3 Actability and Conceptual Modelling ....................................... 284
    16.3.4 Actability and Requirements Engineering ................................. 285
16.4 Lessons Learned and What to Do Next ........................................................ 286
  16.4.1 Actability Design Lessons ................................................................. 286
  16.4.2 The Future of Actability Design ............................................................ 287
  16.4.3 Actability Evaluation Lessons ............................................................... 287
  16.4.4 The Future of Actability Evaluation ....................................................... 288
  16.4.5 Actability Studies Lessons ................................................................. 289
  16.4.6 Future Actability Studies ................................................................. 290
  16.4.7 Actability on Reflection ................................................................. 290
16.5 Summing up the Summary ........................................................................... 291
16.6 Some Final Words ....................................................................................... 291
Index .................................................................................................................. 293
References ......................................................................................................... 297
Part I

Preliminaries
Chapter 1

Introduction

The role of information technology (IT) is changing, as is the way our lives are being affected by that same technology. Information systems are no longer primarily used for storing, retrieving and organizing large amounts of data aimed at informed decision making. Rather, information technology is used in almost every facet of life and has become a naturally integrated part of our daily activities and a concern for our very existence as social beings (Dahlbom and Mathiassen, 1993; Norman, 1998; 2001). Still, our conception of the IT-based information system (IS), from a design point-of-view, has not gone through the same profound changes since those early days in the history of computing when administrative efficiency was at the top of the agenda (Nurminen, 1988; Dahlbom and Mathiassen, 1993; Dahlbom, 1996; Flores, 1998). This state of affairs together with the observation of many failures of IT-based information systems has given rise to a strong imperative to create a better understanding of the nature of such systems and their business use (Orlikowski and Iaccono, 2001).

In order to arrive at such an understanding, this dissertation adopts the position of Hirschheim et al. (1996) who argue that IS development needs to be grounded in theories of social action. According to Weber (1978, p. 4) ‘That action will be called “social” which in its meaning as intended by the actor or actors, takes account of the behaviour of others and is thereby oriented in its course’. This means that social action is not any ‘random behaviour’ but intentional action oriented towards other people’s previous, current or anticipated future behaviour (actions). From a social action-perspective, it is not satisfactory to view information systems as technical black boxes with some social and organizational consequences (Dietz, 2001). We must understand information systems in a deeper sense; information systems should be understood as information technology for social action – as information action systems. This dissertation elaborates on the achievement of such a deeper understanding of the nature of information systems, and in doing so several theoretical and practical implications are pinpointed.

1.1 Information Systems Usage Quality

This dissertation is about a perspective from which information systems\(^1\) are conceived as information technological artefacts intended for business action and communication. This perspective is primarily concerned with qualities of information systems in business use. In that sense it is a ‘usage-centric’ perspective. As we shall see, the perspective has consequences for the design of information systems to avoid deficiencies in the systems and in the interplay between user, system and business, and to promote ‘good’ information system business usage qualities. More specifically, it has consequences for the very conceptualization of information systems and the understanding of IS-related phenomena. It also brings consequences for the development as well as the assessment and evaluation of information systems in different business contexts. A central theme

\(^1\) Throughout the dissertation, the terms ‘(IT-based) information system’ (or ‘IS’ for short), ‘IT-system’, and ‘software artefact’ will be used synonymously.
throughout the dissertation is the importance of considering the business context in order to understand the quality of information systems used. The term business here refers not only to commercial business. Rather, the term reflects action that is goal-directed (purposive) and conforms to an institutionalized set of norms and ‘business rules’ (Stamper et al., 2000). An alternative would be to use ‘organizational context’ since what is referred to is indeed organized action.

When speaking of business usage qualities, ‘quality’ should be understood in a broad sense similar to its definition in ISO 8402: ‘the totality of characteristics of an entity that bear on its ability to satisfy stated and implied needs’ (Bevan, 1995a). This definition highlights that human actors and their ‘stated and implied needs’ in using an entity (in this case an IS) ultimately determine its quality. ‘Utensils play a vital role in our daily round. Throughout our lives we search for tools which are just right. They have to do the job they are intended for, so they have to be carefully conceived. In other words, we strive constantly for “quality”. And when man-made tools work as they should, they encapsulate quality.’ (Ellemann-Jensen, 1997) This is in line with the concept of ‘quality in use’, which Bevan (1995a) defines accordingly: ‘the extent to which a product satisfies stated and implied needs when used under stated conditions’. That is, the quality of an IS should not be considered as a property of the artefact alone but as contingent on its use in actual business contexts. Figure 1-1 depicts this view on software quality.

An IS’s internal quality has to do with static properties of the source code while its external quality has to do with dynamic properties of the system during execution. From Figure 1-1 we can see how the quality in use depends on the external quality, which in turn depends on the internal quality. Accordingly, the internal quality influences the external quality, which ultimately influences the quality in use. This way the usage quality of the IS in a business context becomes the overall goal (design vision) as well as the ultimate effect (result) of any IS design.

The perspective suggested by this dissertation has been made concrete through the concept of information systems actability, defined as ‘an information system’s ability to perform actions, and to permit, promote and facilitate the performance of actions by users, both through the system and based on information from the system, in some business context’ (Goldkuhl and Ågerfalk, 2002). The concept can be understood as an attribute of information systems – information systems possess actability. On the other hand, since it is a concept promoting usage quality, it should be understood as emerging in use. A system can be designed so as to possess external qualities that are likely
to promote actability once the system is in use. This means that what is an actable IS for someone may not necessarily be an actable IS for someone else. Therefore IS actability has been conceptualized in terms of generic properties rather than concrete elements of design. For example, while it is supposed that an actable system makes it effortless to base one’s actions on information from the IS – designing such effortlessness may take on many forms and depends on the particular situation in which the IS is used, including the structure of the business process supported as well as the particular individual users involved.

The perspective and the concept have been formulated in response to identified weak conditions in contemporary views on information systems. These conditions are of different kinds and exist at different levels of abstraction. Nonetheless, they can all be understood as related to two different but partly overlapping paradigmatic foundations for this work, referred to as contextualization and pragmatization of information systems, respectively. These foundations are paradigmatic in the sense that they call for a shift in perspective – a move from one ontological stance to another.

1.2 Contextualization of Information Systems

Many authors have argued for a shift from what can be described as an isolated systems approach towards a contextual approach to information systems design (Goldkuhl, 1992; Kuutti, 1996; Levén, 1997; Beyer and Holtzblatt, 1998; Avgerou, 2001; Holmlid, 2002). In brief, a contextual approach aims at information systems that are usable in the broader context of doing business, not just in the narrower context of humans interacting with computers. That is, a move from focusing internal and external quality alone, towards quality in use (Bevan, 1999).

1.2.1 Usability for Usage Quality

Arguably, one of the most important qualities related to the use of technological artefacts is usability (Shackel, 1984; Nielsen, 1993; Maguire, 2001) or quality in use (Ehn and Löwgren, 1997; Bevan, 1999; 2001) achieved in actual use-situations. Several criteria for assessing the quality of information systems have been proposed over the years. One basic criterion, stressed within the software engineering community, is that the IS should meet the requirements of its users and other stakeholders. Requirements have traditionally been classified as either functional or non-functional (Sommerville, 1996). Functional requirements are concerned with what users possibly can do with a system – the functions, or services it provides. Non-functional requirements, on the other hand, are concerned with restrictions or constraints placed on such a system service. Following this dichotomy in requirements, the usefulness of a system has been described as the combination of its utility (functionality) and usability (Nielsen, 1993).

As pointed out by Grudin (1992) and Bevan (1995a) the distinction between usability and utility implies that it is possible to talk about information systems that are usable but not useful. Bevan (1995a) argues that this seemingly contradictory contention makes sense with a narrow ‘product oriented’ view of usability. With such a view, usability is only concerned with ease of use of the user interface (Holmlid, 2002). Another, broader approach, which is also one of the currently most widely adopted and cited definitions of usability, is that by The International Organization for Standardization in ISO 9241-11 (1998), which identifies usability with the ability to use a product for its intended purposes: ‘the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified
context of use’. This broad approach to usability, which clearly acknowledges the user-context of the system, is similar in concept to quality in use (Bevan, 1999; 2001), defined in ISO/IEC 9126-1 as ‘the capability of the software product to enable specified users to achieve specified goals with effectiveness, productivity, safety and satisfaction in specified contexts of use’ (Bevan, 2001, p. 537).

The three central criteria for usability and quality in use are, as reflected in the above quoted definitions, the effectiveness, efficiency (or productivity) and satisfaction with which users can achieve specified goals. The term ‘effectiveness’ suggests that specified goals are to be achieved with accuracy and completeness (ISO 9241-11, 1998). Effectiveness can be understood as ‘how good a system is at doing what it is supposed to do’ (Preece et al., 2002, p. 14) and is related to the ‘utility’ of the system (Grudin, 1992; Nielsen, 1993), i.e. ‘to the extent to which the system provides the right kind of functionality so that users can do what they need or want to do’ (Preece et al., 2002, p. 16). The term ‘efficiency’ suggests that specified goals are to be achieved with as little expenditure of resources as possible (ISO 9241-11, 1998). The third criterion ‘satisfaction’ suggests that users should feel comfortable with and have positive attitudes towards the use of the system (ISO 9241-11, 1998). The ‘specified context of use’ includes users, tasks, equipment, and the physical environment, where ‘task’ is defined in terms of activities required to achieve a goal (ISO 9241-11, 1998). Apparently, this contemporary notion of usability and quality in use is closely related to the overall goal of designing IT-systems that are useful in their actual context of use (cf. Nielsen, 1993).

According to Levén (1995) as well as, for example, Winograd and Flores (1987) and Goldkuhl and Lyytinen (1982), IS use can be regarded as performing social business action. In accordance with that view, it seems fruitful to speak of a system’s action potential as a combination of action prerequisites (human, social/organizational and technological), performance of actions, and possible intended effects of actions (the things an actor can purposively achieve with the action). The action potential of a system would then correspond to the set of business actions that it supports and could be thought of as an ‘action space’ spanned by the system. This view is in line with Carroll (1994, 1996) who argues that ‘… systems and applications are the set of activities they afford and support’ (Carroll, 1996, p. 267). The action potential would certainly rely on usability in the broad sense as reflected in the ISO definition. To understand an IT-system’s usage quality, it therefore seems important to not only consider a ‘cognitive instrumental rationality, concerned with the evaluation of objective facts’ (Ehn and Löwgren, 1997, p. 308). Since social interests ‘are oriented towards a moral practical rationality, related to evaluations of social action and the norms and practices of social interaction’ (Ehn and Löwgren, 1997, p. 308), we must also consider the use of IT-systems as part of a social action context. Are not the goals of using a system always underpinned by personal as well as institutionalized (inter-personal) values and norms? This is, for example, certainly the position of Stamper (1996) who states that ‘Semiotics [as a theoretical basis for understanding information systems] that excludes norms and attitudes as forms of information would be like physics with the concept of energy but without the concept of mass.’

If so, it seems that an IS cannot possibly be useful if it is not socially accepted. This view relates to the statement by Goldkuhl and Lyytinen (1982) that ‘information systems are social systems only technically implemented’. Nielsen (1993) states that ‘Given that a system is socially acceptable, we can further analyse its practical acceptability…’, which includes its functionality and usability (in the narrow sense). How-
ever, these are not two separate qualities with a causal relationship. Rather, they are so intertwined that separating them is probably not only conceptually inelegant, but in fact misleading.

It has long been claimed that computer systems need to be socially acceptable (e.g. Nielsen, 1993), but such a claim does not necessarily mean that the use of the system is understood from a social action point-of-view. This is important to acknowledge because on one hand, usefulness must be related to social prerequisites, goals, norms, commitments and effects, and on the other hand social acceptability depends on the usefulness of the system. Furthermore, usefulness and social acceptability is not a property of the system being used but a property of the whole use-situation. This context dependency has convincingly been argued as imperative for the design of usable systems (Shackel, 1984; Bevan, 1999; Maguire, 2001) and is also reflected in the ISO definition of usability quoted above. Altogether this means that usage quality (and thus usability) must be studied with social prerequisites, intentions and effects in mind, and that social acceptability relies fundamentally on the usefulness in terms of possible action potential; the set of business actions a system may support. That is, there is a need to consider:

- The system’s action potential, i.e. whether the system, in principle, has the capability of supporting certain business actions.
- The way users can interact with the system in order to perform business actions as well as to interpret previous business actions.
- Users’ intended goals and effects of actions, since it is simply meaningless to speak of actions that have no intended goals and consequences.
- The systems appropriateness in relation to values and established social norms, i.e. it is not sufficient to consider usability and quality in use (including functionality) detached from the business context.

### 1.2.2 The Requirements Gap

When developing information systems it is common to begin with some sort of business modelling in order to get an understanding of the business context in which new systems are to be used (e.g. Jacobson *et al.*, 1994; Bubenko and Kirikova, 1999). What is not so common is to make explicit use of the results of such modelling when defining system requirements, especially not requirements concerning the interaction between humans and IT-systems. Information systems are used to perform and support business actions and therefore models of human-IS interaction ought to be derived explicitly from business models, if the IS is to contribute to the effectiveness of the business.

Of course, there are approaches to information systems development (ISD) that show awareness of the connection between business and information systems (e.g. Jacobson *et al.*, 1994; Jackson and Twaddle, 1997), awareness in the sense that they make use of conceptual business modelling when defining requirements placed on systems’ informational content as well as deriving interaction points from identified business tasks. One thing they seem to miss is that the intentional actions forming business tasks are crucial also when deciding appropriate information content as well as designing user interfaces. A system that is unclear about what business actions it is able to support and maintain information about is most likely unusable.
Many models for interaction design, on the other hand, seem to be too narrow and focus too much on the interaction per se, without appropriately and sufficiently relating it to the business context (cf. Bevan, 1995b). The users’ needs must be judged in the light of the social context in which the users act. This context awareness is emphasized within the field of computer supported co-operative work (CSCW), partly as a reaction against the system focused usability of usability engineering (Löwgren, 1995). However, it requires more than collaboration among the actors to form the business context of a business. In order to design for usage quality we must have a proper understanding of the business, its action structure as well as internal and external actors and agents and their professional language use. Thus, there is a ‘requirements gap’, or at least mismatch, between business modelling and systems modelling (see Figure 1-2).

![Figure 1-2: The requirements gap (adapted from Ågerfalk et al., 1999).](image)

The main concept of this dissertation, information systems actability, can be used as an ‘intellectual tool’ in bridging the requirements gap. The actability solution (in principle) is to (1) regard humans’ interaction with IT-systems as a crucial part of doing business and therefore base its design on business modelling and (2) design business processes with usability in mind. In other words, extend (1) business modelling to incorporate humans’ interaction with IT-systems and (2) usability design to incorporate the design of business processes. That is, in order to create appropriate usage quality of information systems there is a need to broaden the view of IS usage to understand both the business impact of information technology and the business’ impact on information technology.

### 1.2.3 The Cognitive versus the Organizational Approach

How then does one go about deciding required action potential of an IS? One solution to this problem can be summarized by the catchphrase ‘know the user’ (e.g. Nielsen, 1993; Kirby and Rigas, 1999; Faulkner, 2000). It is by acknowledging the user as a human being with capabilities and limitations that user interfaces adapted to the user’s psychological and physiological constraints are possible to create. Consequently, user studies based on methods borrowed mainly from cognitive psychology are important for systems design in this respect. Obviously this is closely related to the narrow approach to usability discussed above. What about functionality, then? According to the approach usually referred to as user-centred design (Gould et al., 1997), it is by studying users’ goals and users’ tasks performed to achieve those goals that required functionality will be properly elicited. A possible problem here is that users’ goals do not necessarily correspond to the goals of the organization they are a part of. Users are at the same time individuals (acting as themselves) and organizational actors (acting on behalf of and as parts of an organization). Assuming that the organization’s goals have
actually been internalized by all members of the organization, a completely user-centred approach might still be intractable when it comes to designing the overall process structures of an organization. The user-centred approach seems to be an ‘individual cognitive approach’ focusing on the individual cognition and practice of a single user (Carroll, 1996) in which high-level goals are eventually arrived at through aggregation of individual goals elicited mainly through user interface prototyping. The left arrow in Figure 1-3 depicts the cognitive approach in which the focus is on actors that perform certain tasks (activities) that eventually make up the business processes of the organization.

An alternative ‘organizational approach’ is offered by various information systems design (ISD) methods. Examples include traditional methods such as Structured Analysis (Yourdon, 1989) as well as more recent ones such as the Rational Unified Process (Kruchten, 1999). These methods approach the problem of defining requirements by starting out with business modelling at a high level and systematically refine produced models during requirements engineering until user requirements have been elicited accordingly. The right arrow in Figure 1-3 depicts the organizational approach in which the focus is on the business processes of the organization and the decomposition of these into tasks (activities) that are carried out by the business’ actors.

As it seems, there would be great benefits if these two approaches could be combined, as suggested by, for example, Activity Theory (Kaptelinin, 1996; Kuutti, 1996). As Carroll (1996, p. 266) puts it: ‘Integrating cognitive and organizational approaches toward human-computer interaction is more than a timely idea; it is essential to an adequate analytical framework for understanding human-computer interaction… Perhaps the more remarkable point here is the apparent non-integration…, given their obvious interdependency’. One way to utilize the strengths of both would be to employ a holistic contextual approach (Goldkuhl, 1992). With such an approach each relevant business context is considered without enforcing decomposition consistency requirements (as with an organizational approach) or risking missing ‘the big picture’ (as with a cognitive approach). Of course, this is closely related to the bridging of the ‘requirements gap’ as discussed above.

Figure 1-3: The cognitive and the organizational approaches.
1.2.4 Information System Users

A further issue concerns who the users really are and how to handle other stakeholders (Preece et al., 1994; Preece et al., 2002; Ehn and Löwgren, 1997). Traditionally, a user (or ‘end user’) is someone directly interacting with a system (or any other kind of product) (Shackel, 1984; ISO 9241-11, 1998). Is the term ‘user’ restricted to those physically interacting with the IS or are other stakeholders affected by the design of the system to be considered as users as well? Levén’s (1995) answer to this question is that the very notion of usage is inappropriate, and he chooses to talk about ‘customer value’ in relation to information systems rather than ‘usage quality’. In doing so he proposes a shift in focus from usage to action. From this perspective, the important issue is not the interaction between humans and computers but between the human-computer constellation and other humans, namely the customers of the organization. Of course, actors in other roles that are affected by actions performed by the human-computer constellation are as important to consider as customers (not all businesses even involve customers). From a social action-perspective it seems that the concept of user must embrace at least someone performing action and someone towards whom the action is oriented: a communicator and an intended interpreter.

1.3 Pragmatization of Information Systems

Pragmatization of information systems refers to the approach taken in this dissertation that seeks to emphasize the pragmatic properties of such systems – to view information systems as systems for action rather than as containers of objective facts. Pragmatics is the branch of linguistics concerned not only with the meaning and structure of language and the concepts used within it, but also with the ways language is used by humans in daily life (Levinson, 1983). The meaning of ‘pragmatization of information systems’ can be understood as a combination of three common information systems fallacies: the usage fallacy, the instrumental fallacy, and the descriptive fallacy. The phrase ‘descriptive fallacy’ was coined by Austin (1962) as a critique against the misconception that language is used only to describe the world. In this dissertation, his idea (as well as his use of the term ‘fallacy’) is explored as a means to emphasize the pragmatics of information systems, which can be regarded as formalizations of the language used within them (Goldkuhl and Lyytinen, 1982).

1.3.1 The Usage Fallacy

A thesis that underpins this dissertation is that contemporary views on information systems and their business usage are too restrictive. Norman (1998) states: ‘I don’t want to use a computer. I want to accomplish something. I want to do something meaningful to me.’ when arguing that ‘you really don’t want to use a computer (even though you think you do)’. This is a good example of what can be referred to as the usage fallacy. People and organizations do business by performing business actions (which by definition are social actions; they are oriented towards the behaviour of others). Computers are used in businesses as tools for performing and coordinating such actions, not merely to be used (Winograd and Flores, 1987; Flores, 1998; Dietz, 2001). Computers are ‘essentially for communication, not computation’ (Flores, 1998, p. 353). This view is in line with Goldkuhl and Lyytinen (1982) who picture an IS mainly as a formalization of the professional language of the business in which it is used.
It is generally agreed that a use-situation must be understood within a larger context than just a user and a computer; we must also consider the task that is being performed by using the computer. This is in line with the broad view of usability as discussed above. Viewing computers (and thus information systems) as communication systems has become commonplace with the widespread use of the Internet and groupware systems. Still, the restrictive usage metaphor is prevalent and ‘we still have a tendency to follow the Cartesian tradition in thinking of communication as the transfer and processing of information’ (Flores, 1998, p. 353).

Shackel (1984) presents a simple and elegant model of a use-situation as consisting of four components: a user, a tool, a task and the environment (see Figure 1-4a). If taking business action rather than usage as the point-of-departure, it seems more natural to talk about actors performing actions through artefacts in a specific business context when referring to IS usage. Figure 1-4b shows this social action-enhanced concept of IS use-situation.

![Figure 1-4: The notion of use-situation according to (a) Shackel (1984) and (b) corresponding actability interpretation – the A³ model (Ågerfalk, 2001).](image)

What the model in Figure 1-4b tries to show is that even though the three binary relations between user-task, user-tool, and task-tool are important to understand, the most important concept is the ternary relation between these three. An actor performs an action directed towards (possibly) another actor by use of an information technological artefact (an IS). Additionally, all these roles must be understood within a particular business context. ‘Business context’ means that the actions performed can be understood as instances of generic business actions structured into business processes following generic business patterns (Winograd and Flores, 1987; Goldkuhl, 1998).

The terminological difference between the models of Figure 1-4a and Figure 1-4b is not just a matter of choice, but reflects a shift in perspective from humans as passive users of information to active performers of business actions (cf. Bannon, 1991).

### 1.3.2 The Instrumental Fallacy

When discussing the use of information systems as the performance of business action, we must recognize that social action is not restricted to instrumental acts (i.e. to the purposive rational behaviour of a single actor and the means he uses to achieve subjective goals). Rather, business actions are possible to relate to goals of different types and require an understanding of social norms (Stamper, 1996; Stamper *et al.*, 2000) in order to be successful. This view of information systems as tools for business action and
communication implies that these are used for performing instrumental acts as well as communicative acts, of which both types are parts of a business context. Consequently, it is not sufficient to describe usage quality only in a subjective and instrumental way. We should also describe the usage of the system from a social and communicative action point-of-view. This implies that we need to take social and not only instrumental goals into consideration, when considering the usage quality of information systems (Eriksson, 2002).

1.3.3 The Descriptive Fallacy
A shift from usage to action also brings with it a criticism of the basic assumption that information systems contain information that is supposed to be used, and indeed updated, by its users. A common characterization of information systems, following the descriptive view, is that they should serve as a mirror or model of reality; a model which users can use to get informed about the state of affairs in some ‘real’ reality (Lyytinen, 1987b). This is a position that has been criticized within the so-called language/action perspective (LAP) as too restrictive a view on the role of information and has been described by Holm (1996) as a ‘technological version of the descriptive fallacy’. As stated above, the phrase ‘descriptive fallacy’ was coined by Austin (1962) and refers to the misconception that language is used for descriptive purposes only. According to speech act theorists Austin (1962) and Searle (1969), language is also used for other things, such as promising, declaring and making commitments. Language is used to bring about changes in the world – we do things by using language (Clark, 1996). A speech act, such as a promise, changes the state of the world in that it creates new social facts such as rights and obligations (Flores, 1998; Searle, 1996). This performative aspect of language must be properly represented and handled in our information systems, both at the interface and in the underlying information structures.

1.4 Aim of Research
The aim of this dissertation is to contribute to a deeper understanding of information systems as tools for business action and communication. Elaborating and reconciling the various views on usage quality as expressed by the paradigmatic foundations mentioned above is the approach taken to achieve this aim. Ultimately this leads to the concept of information systems actability, which is, thus, the main concept under scrutiny.

The overall question for the study is: How can information systems be understood from a social action perspective and what consequences does such an understanding have for the development and evaluation of information systems? A further question, naturally arising from the first is: How does such an understanding contribute to the design of information systems which will actually be more satisfactory?

1.4.1 Detailing the Research Aim
Information systems actability is a concept intended for the understanding of information systems as tools for business action and communication. It can be regarded as summarizing a theory of information systems as information action systems. Such a theory has implications for both the conception and the development of information systems. In the work reported in this dissertation, actability has been developed and
operationalized in three different forms with the purpose of validating its practical usefulness. These are:

Â a system development method referred to as VIBA\(^2\) (see Part III of the dissertation),
Â a method for evaluating the actability of information systems (see Part IV of the dissertation), and
Â a descriptive analytic framework for understanding IS related phenomena (see Part V of the dissertation).

From the point of view of research methodology, to present and elaborate on these three operationalizations constitute subordinate goals that eventually lead to the fulfillment of the ‘ultimate’ aim of the research.

To develop methods to support the development of actable information systems is arguably one of the most important ways to operationalize the concept into directly applicable knowledge. In fact, this is where the actability saga once began. When work on actability started, it served as an articulation and refinement of the theoretical backbone of the already existing system development method VIBA, originally developed by Goldkuhl (1993). While redesigning the method to take advantage of recent and important developments in knowledge, the idea of actability grew and became an important concept beyond merely being a foundation for requirements engineering.

During the work on VIBA it became clear that there was also a need to assess existing systems as well as making formative evaluations. This was when the need for a thorough actability evaluation approach became obvious. Both VIBA and the actability evaluation method represent normative prescriptive action knowledge in the form of method support. This is one important form of action knowledge\(^3\) but not the only one (Goldkuhl, 1999). Creating and applying an analytic framework to guide the analysis of different information systems phenomena is a way of exploring the explanatory power of actability – another important form of theoretical knowledge.

It is important to realize that even though the three operationalizations are treated as three distinct phenomena, they somewhat overlap and are intertwined. For example, some elements of the systems development method VIBA are used in the method for actability evaluation as well. Furthermore, knowledge developed during the work on one operationalization has had substantial impact on the others since they all derive ultimately from the concept of actability. In this dissertation they have been kept separated for mainly two reasons. The first is due to presentation, keeping the discussions focused on one operationalization in turn will hopefully increase readability and understandability of the concepts. The second is due to practical research process concerns; the work has been performed in different project settings, each focused on one of the operationalizations.

As stated above, the aim of this dissertation is to contribute to a deeper understanding of information systems as tools for business action and communication. This is achieved by presenting and elaborating on the three grounding processes used to ground the concept of information systems actability: internal grounding, external theoretical grounding, and empirical grounding. In Chapter 3 this grounding of actability will be described in more detail.

\(^2\) Versatile Information and Business Analysis.

\(^3\) ‘Action knowledge’ refers to knowledge about action, which is intended for action. The concept is further elaborated in Chapter 3.
1.4.2 A Summary of Research Contributions

As indicated above, the research contained in this dissertation exists at various levels of abstraction.

First of all, the dissertation contributes to the understanding of information systems from a social action perspective. This contribution is largely manifested by the concept of information systems actability, which can be thought of as a reconciliation of various views on IS usage quality with its roots in the language/action tradition (Goldkuhl and Lyytinen, 1982; Winograd and Flores, 1987).

At a more concrete level, this understanding, and the concept as such has consequences for the development and evaluation of information systems. Such consequences have been the foundation for a proposed information systems design method called VIBA, which thus constitutes a further contribution. Another contribution is the outlining of an information systems evaluation approach based on the concept of IS actability. This approach is also founded on the design consequences that the proposed way to understand information systems brings.

As a way to get empirical experience of working with the concept of IS actability an analytic framework has been developed, which constitutes yet another contribution. Finally, when applying the framework, two further contributions have been made. These are constituted by characterizations of the Internet-based information system (a particular form of IS constructed using Internet-technology) and the local electronic marketplace (an Internet-based practice that concerns consumer-oriented business activities performed by actors using a digital (IT-based) as well as a physical channel for customer interaction), from the perspective of actability.

1.5 Demarcations

The focus of this dissertation is on information systems’ qualities related to the business use of such systems. This focus implies a limitation of the scope of the dissertation. That is, it will not specifically address, but might still touch upon, residual issues concerning, for example, maintenance and IS economics. It is however important to be aware that even though the focus is on use, that focus has repercussions on both development and evaluation, as well as on the very conceptualization of information systems.

Above, the term business was designated to refer to organized collaborative behaviour aimed at some articulated and agreed upon goals. However, in this dissertation, the actability concept will not be discussed in usage settings such as computer games and home electronics, even though it would probably make sense outside the realm of business and information systems as well.

Choosing to view IT-systems as tools for business action and communication may in itself seem to imply a demarcation in scope. In this dissertation, the focus is on IT-systems that are used in business settings. This means that the systems and the actions taken through, and by means of, the systems are related to at least two actors. This does not mean that we are restricted to pure communications systems or systems to directly support collaborative work. Rather, all IT-based information systems are related to a social context as they are products of human action used in human action (de Souza, 1993). However, the focus in this dissertation is on IT-systems that are used (a) to automatically perform action on behalf of some actor directed towards another actor, thus acting as an agent; (b) by some actor to interactively perform action through the system directed towards another actor, possibly on behalf of someone else; and (c) to
inform the performance of manual action by some actor directed towards another actor outside of the system. These three types of IS usage (a, b, and c) constitute three different types of IS use-situation, which are elaborated further in Chapter 4.

1.6 A Note on Related Research

Related research will be considered throughout the dissertation, and also summarized in Chapter 16. This section is intended to give a rough overview and to frame the dissertation in relation to what other people have done. This is important since the social action perspective proposed in this dissertation is not a completely new concept and several scholars have argued for similar views. Most convincingly, such a perspective has been argued by Winograd and Flores (Winograd and Flores, 1987; Winograd, 1988; Flores, 1998) and several other authors working with linguistic and semiotic approaches to information systems seem to acknowledge the fundamentals (e.g. Flores and Ludlow, 1980; Goldkuhl and Lyytinen, 1982; Auramäki et al., 1988; Denning and Medina-Mora, 1995; Holm and Ljungberg, 1996; Stamper et al., 2000; Dietz, 2001).

A common denominator in these authors’ work is that it is too restrictive to view IT-systems and computers as just another form of media. IT-systems should rather be viewed as complex semiotic (communicative) systems that intertwine with users’ social life in fundamentally different ways than traditional communication devices such as radio and telephones. These approaches do not, however, go into the details of information systems and specifically not into usability issues. Rather, their focus is on business and communication modelling in relation to information systems development at a fairly abstract level.

Once again, there are many other related areas and approaches that will be addressed throughout the dissertation.

1.7 Dissertation Structure and Research Setting

This dissertation is a result of several years of collaborative research efforts and several people have contributed to its content. Active participants in the work reported, in addition to the author, have been Göran Goldkuhl and Stefan Cronholm, Linköping University; Anders Hjalmarsson, University College of Borås; Emma Eliason, Fredrik Karlsson, Per Oscarsson and Johan Pettersson, Örebro University; Jonas Sjöström, Jönköping International Business School; and Owen Eriksson, Dalarna University. In addition, Pär Fahlén, formerly Örebro University nowadays at Infogrator Örebro AB, participated initially.

The dissertation is based on a selection of papers. It is, however, not organized strictly as a collection of articles with a summary. Rather, the different articles have served as a background and are incorporated in the text throughout the dissertation to create a hybrid between a collection of papers and a monograph. Such an approach has the advantage that central ideas and concepts can evolve over time and comments from, for example, conference participants can be reflected in the dissertation while it can still benefit from the continuous quality assurance imposed by refereeing (and indeed, co-authoring). The scientific work upon which this dissertation is based is shown in Table 1-1.

This dissertation may be considered as an extension of the author’s licentiate thesis (Ågerfalk, 1999b). Therefore, some parts of the dissertation, and some of the results reported in it, overlap with this previous publication. Specifically, papers 7, 9, 13, and 15 (see Table 1-1) were a basis also for the licentiate thesis.
<table>
<thead>
<tr>
<th>No.</th>
<th>Author(s) and Year</th>
<th>Title and Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.</td>
<td>Ågerfalk P J (2003)</td>
<td>Nine Principles for Actable Systems Design, Accepted for the 10th International Conference on Human-Computer Interaction (HCI International 2003), Crete, Greece, 22–27 June 2003. This work is the copyright of Lawrence Erlbaum Associates, Inc. It has been included in this dissertation by kind permission of the publisher.</td>
</tr>
</tbody>
</table>
Introduction


The dissertation is divided into six parts. The remainder of this section presents the different parts and how the scholarly works listed in Table 1-1 has been used within them (references to the works listed in Table 1-1 is shown within square brackets). In addition to the major references to the scholarly works listed in Table 1-1 and discussed below, some fragments, ideas and arguments from previous work have been scattered and used in various parts of the dissertation. One can, for example, find fragments from Cronholm et al. (1999) and Ågerfalk (2001) in the dissertation.

1.7.1 Introducing Part I of the Dissertation

Part I, *Introduction* (Chapters 1 to 3), introduces the work and the field of study. This part of the dissertation, which you are reading now, sets the baseline for the work by presenting the background and the research questions addressed (Chapter 1), the field of study (Chapter 2), and the overall research approach used in the work (Chapter 3). Section 2.3 is an extension of [13] a joint paper by the author of this dissertation and Stefan Cronholm. The rest of the section consists mainly of new material.

1.7.2 Introducing Part II of the Dissertation

In Part II, *Information Systems Actability* (Chapters 4 to 7), the focus is on internal grounding (mainly in Chapters 4 and 5) and external theoretical grounding (mainly in Chapters 6 and 7). In Chapter 4, the concept information systems actability and its theoretical background is presented and elaborated. The chapter is based on [16], a book chapter by the author of this dissertation and Göran Goldkuhl. In Chapter 5 one important aspect of actability, the action-elementary message is discussed in more detail and related to the use of information systems as tools for business action and communication, as well as to traditional Langeforsian (1973; 1995) ‘infologial’ sys-

---

4 The concepts of internal, external theoretical, and empirical grounding as used in this section are described in Section 3.2.2, p. 40 ff.
Information systems actability

tems theory. The chapter is an extended version of [1], a paper by the author of this dissertation. Chapter 6 positions the proposed view of information system in relation to conceptual modelling as a systems development activity. The chapter is based on [5], a joint paper by the author of this dissertation and Owen Eriksson. Finally, Chapter 7 positions the proposed view in relation to the concept of usability. The chapter is based on [6], a joint paper by the author of this dissertation and Owen Eriksson in which central ideas of actability and results from Eriksson’s doctoral dissertation (Eriksson, 2000) are combined.

1.7.3 Introducing Part III of the Dissertation

Part III, Actability Design (Chapters 8 to 10), is devoted to the first of the three operationalizations of actability and hence mainly to empirical grounding. More specifically, the focus is on the systems development method VIBA concerned mainly with business and information modelling as a part of requirements engineering. Active participants in the work on VIBA have been, in addition to the author of this dissertation, Göran Goldkuhl and Stefan Cronholm. Additionally, Pär Fahlén participated initially in what is referred to as Application 1 in Chapter 8, the chapter that describes the Actability Design work in detail. In this chapter, Section 8.2.3 is based on an excerpt from [15]. The VIBA method is presented in Chapter 9. The chapter conflates and extends three conference papers: [7], [9] and [15]; and a book chapter: [8]. Chapter 10 presents the evolution of VIBA and lessons learned from the empirical application of the method. The chapter contains new material and some parts from the author’s licentiate thesis.

1.7.4 Introducing Part IV of the Dissertation

Part IV, Actability Evaluation (Chapters 11 to 13), concerns the second of the three operationalizations of actability. The focus is on a method for in-context evaluation of information systems. Active participants in this work on actability evaluation have been, in addition to the author of this dissertation, Göran Goldkuhl, Stefan Cronholm, Emma Eliason, and Jonas Sjöström. Chapter 11 describes the parts of the research on actability evaluation in which the author of this dissertation has been involved. Some complementary work (not explicitly reported in this dissertation) has been carried out by the other participants, and is reported elsewhere (e.g. Cronholm and Goldkuhl, 2002; Sjöström and Goldkuhl, 2002). The evaluation method is outlined in Chapter 12 and empirical lessons learned are presented in Chapter 13. Chapter 12 is basically a considerably extended version of [12] – a joint paper by all of the aforementioned participants. Parts of this extension are reported in three papers by the author of this dissertation and Eliason: [2], [4], and [14]. These three papers also form a basis for the chapter. Chapter 13 contains mostly new material but incorporates some parts of [12], and Section 13.3.3.1 is based on [14].

1.7.5 Introducing Part V of the Dissertation

Part V, Actability Studies (Chapters 14 and 15), concerns the third and final operationalization of actability in this dissertation. This part is devoted to two different studies in which a descriptive analytic framework based on actability has been developed and used. The first study (Chapter 14) concerns a characterization of the Internet-based information system from the perspective of information systems actability combined with the so-called ‘semiotic framework’ (Stamper, 1996). The second study (Chapter 15) concerns a characterization of the Local Electronic Marketplace from the perspec-
tive of information systems actability combined with a comprehensive notion of practice (Goldkuhl and Röstlinger, 1999). Both studies aim to provide concrete examples of actability concepts and to establish that these concepts are useful to discuss these types of system. That is, to present good, empirically justified reasons as arguments for the concept of actability, outside of the realm of systems development and evaluation.

Chapter 14 (the first study) is based on an amalgamation of two papers, [10] and [17], by the author of this dissertation, Anders Hjalmarsson and Fredrik Karlsson. Although the study was a truly joint effort, the author of this dissertation did most of the work on the analytic framework used in the study.

Chapter 15 (the second study) is based on a journal paper by the author of this dissertation and Johan Petersson [11]. During this study, Petersson performed the data collection, while the analysis, theorization and presentation was of a truly cooperative nature. As in the first study, the author of this dissertation did most of the work on the analytic framework. This work is closely related to, and builds upon, Peterson’s work on local electronic marketplaces (Petersson, 2001a; b), which was also the main topic for his licentiate thesis (Petersson, 2002).

1.7.6 Introducing Part VI of the Dissertation

Part VI, Conclusion (Chapter 16), concludes the work and points at some possible future research directions. The chapter contains mainly new material. Section 16.3.1 is based on a poster summary [3] by the author of this dissertation and Stefan Cronholm.

1.8 Intended Audience and Reading Instructions

This dissertation has been written with two large target groups in mind. Research colleagues within the fields of requirements engineering, information systems and human-computer interaction constitute the first group. Practitioners, struggling with the kinds of problems addressed by the dissertation, constitute the other. This latter group includes systems developers and user interface designers (usability engineers, et cetera). With the varying theoretical and formal backgrounds this audience probably represents, trying to write clearly and concisely without being either too inconsiderate or too prosaic has been a considerable challenge which the author hopes he has surmounted.

Specifically, the presentation of the material assumes that readers possess a general knowledge of systems development and are used to following discussions and argumentation at a fairly abstract level. Existing, well documented, formalisms and notations are not described in detail. Instead references are given to relevant sources. In particular the presentation of VIBA (Part III) assumes familiarity with the Unified Modelling Language (UML) and its notation for class models (Class Diagrams) and state models (Statecharts); see, for example, Booch et al. (1999) for an introduction.

The theoretical discussions in this dissertation build quite strongly on language philosophical and linguistic traditions. Probably, a basic understanding of speech act theory (Austin, 1962; Searle, 1969; 1979), communicative action theory (Habermas, 1984) and, possibly, semiotics (e.g. Innis, 1985) and specifically pragmatics (e.g. Levinson, 1983) will make the dissertation easier to comprehend.

When reading the dissertation it is possible to follow one or more tracks depending on one’s particular interest in the work. Table 1-2 shows a selection of such tracks that suggests which chapters might be of most interest from different perspectives likely to be identified within the intended audience. Of course, it is highly recommended to read the whole dissertation to get the complete picture.
Table 1-2: Suggested reading tracks through the dissertation (Â symbolizes chapters that are particularly recommended and Ɣ symbolizes chapters that might be of interest).

<table>
<thead>
<tr>
<th>Track</th>
<th>Part I</th>
<th>Part II</th>
<th>Part III</th>
<th>Part IV</th>
<th>Part V</th>
<th>Part VI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complete track</td>
<td>Á Â Á Á Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fast track</td>
<td>Á</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Research methodology track</td>
<td>Á Â</td>
<td>Â</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HCI track</td>
<td>Á Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IS method track</td>
<td>Á Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E-commerce track</td>
<td>Á Â</td>
<td>Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Semiotic and L/A track</td>
<td>Á Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Theoretical track</td>
<td>Á Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Chapter 2

Field of Study

The aim of this dissertation was formulated in Section 1.4 in terms of ‘contributing to the understanding of information systems’. This understanding has been formulated and packaged in the form of the concept of information systems actability. The concept can be understood as representing a quite extensive theory of information systems as information action systems. As described in Section 1.4.1, this theory has been concretized by way of three different operationalizations concerned with actability design, actability evaluation, and with the study of IS related phenomena. The methods for Actability Design and Actability Evaluation represent normative knowledge in the form of method support, while the Actability Studies focus on actability as an illustrative theory for scientific enquiry. To put the work in a scientific and practical context, this chapter expands upon the notions of information systems theory and specifically on the concept of method as it is being used in information systems development (design and evaluation). The use of theory in scientific enquiry will be elaborated in Chapters 3, 14 and 15. First, however, something will be said about the scientific discipline within which the work has been performed, in order to set up a general frame and motivation for the work.

2.1 Scientific Abode

This work has been carried out within the scientific discipline that is nowadays usually referred to as informatics. In Sweden, informatics is considered part of the social sciences and is concerned with the use and development of information technological artefacts, such as information systems (IT-systems), within a social context. Informatics has been defined by the Swedish professor Pelle Ehn (1995) as ‘the art and science of designing IT artifacts’. Another Swedish professor Bo Dahlbom (1996) expands upon this notion by referring to it as ‘a discipline tracking (leading) the development of information technology, with the ambition to put that technology to good use, acting both on the technology and on the organization of its use’, and points out that even though the subject matter of informatics is IT use, ‘…this interest is design oriented’. Thus, a central concept for informatics is that of design. A third Swedish professor Göran Goldkuhl (1996) reflects upon these definitions and adds that design must be understood as both a process (activity) and as the form of the created result. He also points out that not everything can be treated as intentional design and therefore chooses to include change as the more generic concept. Thus, informatics is about people’s activities that bring about change. The change takes place at the level of information technology, as well as at the level of that technology’s use in relation to people, organizations and society.

To sum up, informatics is characterized by an interest in use-oriented design of IT-systems. The design interest concerns both the process of design and its result in terms

---

5 At Linköping University where this dissertation is put forward, the graduate study subject is for historical reasons called ‘information systems development’.
of properties of the designed artefact. The design-process interest concerns people’s work with constructing IT-systems. This interest is manifested in studies of ways of working, methods, and tools for the development, evaluation, maintenance, *et cetera*, of such systems. The design-as-property interest focuses on the form and function of IT-systems in relation to the needs of system users and other stakeholders. This interest is manifested in studies concerning the use of IT-systems and theorization of the IT-system phenomenon *per se*. Such studies may include the construction of IT-systems, but then the design process itself may not be in focus. A characteristic of informatics is that both design aspects are founded on IT-systems actual or possible use in business life, private life and society as a whole. This dissertation concerns both aspects of design and focuses on IT-systems for business use.

The epistemological tradition within Scandinavian informatics is based on an interest in exploring and analysing foundational concepts, such as ‘data’, ‘information’ and ‘communication’. This tradition goes back to the founder of the discipline in Sweden, Börje Langefors, who also was the first professor of this discipline (then by the name of Information Processing – Especially the Methodology of Administrative Data Processing). His work, most remarkably the seminal ‘Theoretical Analysis of Information Systems’ (Langefors, 1973), first published in 1966, has paved the way for placing IT-system use and development within a firm theoretical discourse. This dissertation builds on that foundation by focusing on the further development of the theoretical understanding of information systems as tools for business action and communication.

2.2 Information Systems Theory and This Work

Different authors have pictured and discussed information systems development in terms of *theory* (e.g. Langefors, 1973; Hirschheim et al., 1996), *perspective* (Lyytinen, 1987a; Winograd, 1988; Goldkuhl, 1994; Opdahl, 1998), *tradition* (Näslund, 1996), and *paradigm* (Nurminen, 1988; Hirschheim and Klein, 1989).

The meaning of the term ‘theory’ is not always clear and it is used in many different ways in different contexts. The notion of theory is, according to Dahlbom & Mathiassen (1993), really a romantic notion ‘stressing the importance of going beyond the observable phenomena to deeper, hidden layers of reality, in order to define concepts and identify general laws, in terms of which the chaotic flux of observable facts can be systematized and explained’ (Dahlbom, 1996, p. 41). According to Dahlbom, the term is often ‘deromanticized’ to simply mean ‘an alternative conceptual schema to the one used by common sense, but the ambition remains the same, namely to bring order and sense to a complex world’ (Dahlbom, 1996, p. 42).

Thus, in its most general form, a theory can be thought of as a set of definitions and axioms, and theorems derived from these. Euclidean geometry is an example of such an ‘ideal’ theory. In this dissertation, the more generous, and also ‘deromanticized’, interpretation of the concept of theory is used. A theory of information systems is here thought of as a set of properties that constitutes what is meant by information and information systems. This concept of theory is seemingly similar to the concept of perspective (Dahlbom and Mathiassen, 1993) or frame of reference (Orlikowski and Gash, 1994). According to Opdahl (1998), a perspective is a mental representation of the world in terms of interesting/purposeful things and a selection of interesting/purposeful properties of those things. Opdahl stresses that a perspective is subjective and that different people might see different things when looking at the same piece of reality. Thus, a theory of information systems is an externalized, possibly inter-
subjective, conceptualization of interesting properties of information, information systems, and information systems development. Put another way, a theory emphasizes certain kinds of properties and directs attention to particular kinds of things and properties of those things. When perceiving a phenomenon based on such a theory, a perspective is applied.

When several people share the same or similar perspectives and work according to a common theoretical basis, their ways of working eventually become institutionalized into a tradition. People within a tradition thus share a common set of values and conceptualizations and ‘take certain aspects more or less for granted’ (Näslund, 1996). Examples of traditions that Näslund mentions are the information system methods tradition, the software engineering tradition, and the human-computer interaction tradition. Näslund further claims that traditions are similar to scientific paradigms. However, traditions differ from paradigms since many people work across the borders of two or more traditions, that is, what Näslund (1996) refers to as boundary spanners, and thus are members of many traditions. Nurminen (1988) makes a more accurate interpretation of paradigm in the discussion of the system theoretical paradigm, the socio-technical paradigm and the humanistic paradigm. The emergence of each of these paradigms follows the kind of revolutionary change scientific paradigms are supposed to follow according to Kuhn (1970).

Tradition and theory are not equivalent classification schemes. Within a tradition there might exist different theoretical assumptions that form groups within a tradition. Groups might even span several traditions, just like individual tradition members. Within a tradition there are assumptions (explicit or implicit) about interesting problem domains and desirable types of solutions to problems. Different traditions might however regard the same or similar problems as interesting but approach them in different ways. For example the field of requirements engineering is highly relevant to both the information system method tradition and the software engineering tradition.

Table 2-1 shows some examples to clarify the relations and distinctions between tradition, theory and perspective.

Table 2-1: Examples of concerns and possible answers at the levels of tradition, theory and perspective.

<table>
<thead>
<tr>
<th>Level</th>
<th>Example Concern</th>
<th>Possible Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tradition</td>
<td>What problems are important to study?</td>
<td>Development and use of information systems</td>
</tr>
<tr>
<td></td>
<td>What are desirable kinds of solutions?</td>
<td>Methods for developing information systems</td>
</tr>
<tr>
<td>Theory</td>
<td>How should problem domains be conceptualized and explained?</td>
<td>Information systems as information action systems</td>
</tr>
<tr>
<td>Perspective</td>
<td>How should this situation be interpreted based on the theory used?</td>
<td>An actor performs a business action through the IS</td>
</tr>
</tbody>
</table>

Using the classification introduced by Näslund (1996), this work has been carried out within the information system method (ISM) tradition, but with inspiration from the software engineering and the human-computer interaction traditions. Specifically, the work has been performed based on the so-called language/action perspective, which thus forms a ‘group’ within the information system method tradition. The language/action perspective is not exclusive for the ISM tradition, but has been applied, for
example, also within the software engineering (SE) tradition (e.g. Graham, 1998) and the human-computer interaction (HCI) tradition (e.g. Winograd, 1988). Figure 2-1 shows how this work relates to these three traditions and the language/action perspective.

As shown in Figure 2-1, the three traditions mentioned relate to and influence each other in other ways than what is treated in this dissertation. This work is delimited to ideas and theories within the influencing traditions that conform to the adopted language/action perspective.

2.3 The Concept of Method in Systems Development

The concept of method has been discussed through several decades (Bubenko and Källhammar, 1970; Checkland, 1981; Goldkuhl and Lyytinen, 1982; Jayaratna, 1994; Harmsen, 1997; Russo and Stolterman, 2000; Avison and Fitzgerald, 2003). Information systems development (ISD) methods are often used during development and

---

evaluation of information systems in order to guide and support the work process, i.e. ISD methods can be thought of as normative conceptualizations that direct method users’ attention to certain kinds of phenomena. Hence methods are created and used to support ISD actors in performing different tasks aiming to reach some goals (Ågerfalk and Ahlgren, 1999).

However, the meaning of the concept of method is not always clear. When studying different theories about methods or looking into different practical manuals it is obvious that several definitions of the concept of method exist, and there are also new emerging concepts. Furthermore, there exist several related concepts – often with ‘method’ as a prefix and a following noun (method alliances, method components, method fragments).

When looking more closely at different concepts, one can identify that there are different concepts (and terms) used for the same phenomenon and also the same concept (and term) for different phenomena. The definition of ‘method’ chosen in this thesis originates from several researchers who have all contributed to the understanding, and to the development, of the concept of method. The aim of this section is to discuss different method concepts and explain how these are related to each other to serve as rationale for the use of the concept of method in this dissertation.

2.3.1 Method versus Methodology

Jayaratna (1994) defines ‘method’ as ‘an explicit way of structuring one’s thinking and actions. Methodologies contain model(s) and reflect particular perspectives of “reality” based on a set of philosophical paradigms. A methodology should tell you “what” steps to take and “how” to perform those steps but most importantly the reasons “why” those steps should be taken, in a particular order’. As we can see Jayaratna uses the term ‘methodology’. Methodology is a Greek term meaning the study of methods. The Oxford English Dictionary defines methodology as ‘the study of systematic methods of scientific research’. Jayaratna justifies the use of methodology claiming: ‘however, the term methodology is pragmatically well established within the field of information systems to mean the same as method’.

Another example of the use of ‘methodology’ synonymously with ‘method’ is that of Stamper (1988), when stating that ‘I use the term “methodology” under protest bowing only to customary usage. It would be better, as in philosophy of science, to speak of “methods” when referring to specific ways of approaching and solving problems, and to reserve “methodology” for comparative and critical study of methods in general; otherwise this vital field of study is nameless’.

A third example of the misuse of the term methodology is by Brinkkemper (1996, p. 276), who states that ‘the misuse of the term methodology standing for method is a sign of the immaturity of our field, and should consequently be abandoned’. Brinkkemper (1996, pp. 275-276) further defines method as an ‘approach to perform a systems development project, based on a specific way of thinking, consisting of directions and rules, structured in a systematic way in development activities with corresponding development products’.

Another view on this matter is presented by (Bennett et al., 2002) who proposes to use the term ‘method’ when speaking of the specific approach taken in a particular systems development effort and using ‘methodology’ for the generic description of the approach used. In this view a methodology may be tailored and used in many projects, each applying its own method based on the methodology. This latter use of the term
‘method’ corresponds to what Brinkkemper (1996, p. 277) terms ‘situational method’: an information systems development method tuned to the situation of the project at hand.

A third definition of the concept of method is that by Röstlinger & Goldkuhl (1994). Their definition reads: ‘methods are prescriptions for human actions and methods are normative and guide the ISD process’ (author’s translation to English).

Another definition from Checkland (1981) runs: ‘a methodology will lack the precision of a technique but will be a firmer guide to action than a philosophy. Where a technique tells you “how” and a philosophy tells you “what”, a methodology will contain elements of both “what” and “how”’. Checkland also uses the term methodology to mean method. Furthermore, Checkland uses the concept ‘technique’. Exactly what he means by technique is not defined, but he is probably referring to a diagramming technique, such as data flow diagramming or entity-relationship diagramming, which, according to Brinkkemper (1996, p. 276), however, defines ‘technique’ strictly as: ‘a procedure, possibly with a prescribed notation, to perform a development activity’.

When examining the method definitions above, it is clear so far that the term ‘methodology’ is often used when what is actually referred to is ‘method’. Method is descended from the Greek language, meaning ‘way of investigation’. The meaning of ‘method’ seems to answer the question how ISD shall be performed and ‘methodology’ is the study of methods. Furthermore, when an ‘ideal typical’ method is followed in practice (possibly in a ‘tuned’ way), it is possible to talk about a ‘situational method’.

### 2.3.2 Other Method-Related Concepts

#### 2.3.2.1 Method Types

Nilsson (1991) distinguishes between ‘method’ and ‘method type’ (in Swedish: ‘metodik’, there is no corresponding established English term except for ‘methodology’, which above was designated to mean the study of methods). Nilsson defines a method type as a general concept (a type of methods) and a method as a specific concept (an instance). In other words, a method is a concrete representation of a method type. Following Nilsson’s definition, object orientation is an example of a method type and the Object Modeling Technique (OMT) is an example of a method of that type.

#### 2.3.2.2 Method chains and alliances

Fåhraeus (1986) talks about ‘method chains’ as consisting of several methods linked to each other where the result from a method used in an earlier step are used in a following step.

Nilsson (1999) has further developed the concept of method chain. Nilsson’s definition runs: ‘Integration of methods between different levels of development work. This approach to combine methods is a kind of vertical integration’. Nilsson points out that there are several (abstraction) levels of development work. For example there could be a higher level dealing with conceptual modelling, followed by object modelling performed in a lower level. The object model can in turn be used when defining the database schema. Nilsson’s division into a vertical integration makes sense when he also introduces the concept ‘method alliance’.

A method alliance is an integration of methods within the same level of abstraction. This is a horizontal integration of methods. Nilsson states that alliances are motivated by the need ‘to tackle several problems or perspectives in concrete situations’.
That is, method alliances cover several aspects of a problem domain at a specific abstraction level. A method usually covers several phases and aspects within ISD. A ‘method’ thus addresses ISD into a congruent whole. What Fåhraeus and Nilsson seem to mean by ‘method’ is more accurately interpreted as method fragments or method components (defined below).

2.3.2.3 Perspective

Another method-related concept is ‘perspective’. A perspective is a manifestation of theory of how ISD shall be performed (Nurminen, 1988). Such a theory shall be normative, explanatory and classifying. Mathiasen (1982) defines ‘perspective’ as a conceptual abstraction of a view of a specific phenomenon (cf. Section 2.2). Jayaratna (1994) says, ‘methodologies ... reflect particular perspectives of “reality” based on a set of philosophical paradigms’. …

In other words, the method constructor’s perspective is based on how he or she perceives the world. The method constructor’s values and beliefs thus influence the method user when performing ISD. A perspective implies, for example, what primitives to use and these primitives in turn influence method users (Ågerfalk and Åhlgren, 1999). The character of the influence can be either governed or supported. The perspective is not necessary made explicit in the method. The method constructor’s perspective is often implicit and taken for granted. One can say that a method is always based on a perspective with which follows, for example:

- principles,
- values,
- conceptions,
- experiences,
- categories, and
- definitions.

To sum up, the perspective, explicit or implicit, influences the method user in one way or another. We can distinguish between internal and externalized perspective of a method creator (or any human being). The parts of a person’s conception of the world that are hard (or even impossible) to externalize constitutes that person’s internal perspective. The externalized perspective, on the other hand, is constituted by intersubjective values, beliefs, et cetera, to which the method creator adheres. Examples of existing externalized perspectives in ISD are business orientation, object orientation and user-centeredness.

2.3.2.4 Framework/Model

Another method-related and sometimes confusing term is ‘model’. What do we actually mean when talking about models? According to Yourdon (1989), a model is used to ‘highlight, or emphasize, certain critical features of a system, while simultaneously de-emphasizing other aspects of the system’. Examples of classical notations to express models are data flow diagrams and entity-relationship diagrams (Yourdon, 1989). Rumbaugh et al. (1991) define ‘model’ as ‘an abstraction of something for the purpose of understanding it before building it’.

Jayaratna (1994) defines ‘framework’ as a static model, which provides a structure to help connect a set of models or concepts. Goldkuhl (1991) defines ‘model’ as a structure for the ISD process that defines and delimits specific areas within ISD that form related phases. A model answers the question of what is to be done but not how it
should be done. Examples of such models are the classical Swedish SIS/RAS model and the LOGIC model.

What makes the definitions above confusing is that they refer to different domains. When examining Yourdon’s definition, it is obvious that he is referring to a model of an IS. The same goes for Rumbaugh et al. However, when Goldkuhl talks about models he is referring to a model of the ISD process. In other words, they are using the same term but referring to different concepts. Jayaratna (1994), similarly to Goldkuhl, refers to the ISD process whilst the others refer to the product of such a process. Furthermore, in the Rational Unified Process (Kruchten, 1999) the model referring to the structure of the process is referred to as a process ‘architecture’. In Röstlinger & Goldkuhl (1994) the framework concept is also used as a synonym for model.

To avoid confusion it seems better, as Jayaratna does, to use the concept of framework when referring to the ISD process, or use the term ‘development process model’ to indicate that we are talking about a model of a development process.

The concept ‘framework’ is well defined in the software engineering tradition but not fully applicable in the information systems tradition. One definition from the software engineering tradition runs (Öberg, 1998): ‘A framework is a generic design solution to a certain problem or a certain domain. The framework describes the different design elements involved in the solution, as well as their relations’. If one changes the term ‘design solution’ to ‘ways of performing ISD’ and the term ‘design elements’ to ‘phases’ the definition becomes similar to Röstlinger & Goldkuhl’s (1994) definition of ‘framework/model’.

2.3.2.5 Method Components and Method Fragments

A method can be perceived as a ‘whole’ consisting of different ‘parts’. Therefore we also need a concept for the parts of a method. During the last decade, concepts such as ‘method components’ (Röstlinger and Goldkuhl, 1994) and ‘method fragments’ (Harmsen, 1997) have been proposed to talk about method parts. A reason for this is a move from viewing methods as monoliths to a generic flexibility (Röstlinger and Goldkuhl, 1994) suited for situational method engineering (Harmsen, 1997; Brinkkemper et al., 1999).

The concept ‘method fragment’ is defined by Harmsen (1997) as ‘… a description of an IS engineering method, or any coherent part thereof’. From this definition, a complete method, for example OMT, is a method fragment and so is any single concept used within OMT, as for example ‘object’. To sort this out, a method fragment is said to reside on a certain layer of granularity, of which five are possible: method, stage, model, diagram, or concept. Thus, ‘object’ resides on the concept layer and ‘OMT’ on the method layer. Furthermore, a method fragment is either a process fragment or a product fragment. Process fragments represent the activities, stages, and cetera, that are to be carried out and product fragments represent deliverables, diagrams, and cetera, that are to be produced, or that are required, during development.

Röstlinger & Goldkuhl (1994) view methods as constituted by exchangeable and reusable components. Each component consists of descriptions for ways of working (a process), notations, and concepts. A process describes rules and recommendations for the ISD and informs the method (component) user what actions to perform and in what order. Notation means semantic, syntactic and symbolic rules for documentation. Concepts are categories included in the process and the notation. A method component can be part of a method chain or a method alliance. Concepts and notation thus together constitute what is sometimes referred to as a modelling language, such as the UML.
(Booch et al., 1999). A method component or fragment can also be used separately and independently from other components. Each method component addresses a certain aspect of the problem at hand. Examples of method components are ‘use case analysis’ (Jacobson et al., 1992) and ‘finding classes & objects’ (Coad and Yourdon, 1991), which are both parts in a whole (a method).

Thus, a method component can be thought of as the smallest meaningful assembly of method fragments to address a certain aspect of a problem (Brinkkemper et al., 1999) and consists of product fragments (notation), process fragments (process) and concept fragments (concepts) used in the other two types of fragments. Note that a method component per se is a method fragment at some intermediate layer of granularity.

2.3.2.6 Co-operation Forms

The Scandinavian tradition of performing ISD often means that several actors are involved in the ISD process. Hägerfors (1994) describes the ISD process as a group process with actors who interact, discuss, learn, agree, disagree and argue. Several research reports argue for strong user (business actor) participation. This means that methods also should support co-operation forms. According to Goldkuhl et al. (1998) co-operation forms describe how different persons interact and co-operate when performing method guided work. Co-operation also has to do with roles and division of work. One can say that co-operation forms deal with the meta-question of who is to ask the questions during ISD (Goldkuhl et al., 1998). Examples of co-operation forms are brainstorming sessions, interviews and modelling sessions.

Harmsen (1997) distinguishes between two different domains that are in focus during ISD. Some ISD activities belong to the target domain and some to the project domain. The target domain consists of activities directly addressing ISD and the project domain consists of activities addressing management of ISD. Co-operation forms thus seem to belong to the project domain.

2.3.3 Relating the Concepts

Goldkuhl et al. (1998) relate some of the main concepts discussed above and present them as depicted in Figure 2-2. In their view, information systems work can be conceptualized as a set of questions. In this view, the ISD process can be understood as the series of questions to ask during ISD. The concepts used answer the meta-question: what to talk about when questioning, and the notation dictates how to express answers to the questions. The questions asked are related to each other in the framework of the method (the process model) and suggested co-operation forms recommend who should ask and who should answer the questions. Finally, the underlying perspective suggests what it is that is important to ask questions about.

The model suggested by Goldkuhl et al. (1998) gives a good overview of the concept of method, but several important concepts from the analysis so far are missing. In this section, the analysis will therefore be taken further and include also the other concepts discussed above. The resulting view of the concept of method is depicted in Figure 2-3.

From the analysis presented we can observe that there is a need for a distinction between ‘method’ and ‘methodology’. As mentioned above, the relationship between

---

7 Cf. the distinction made between Core Process Workflows and Core Supporting Workflows in earlier versions of the Rational Unified Process (Kruchten, 1999).
A method implies a perspective. As discussed previously, a perspective is either explicit or implicit in the method (or methodology for that matter).

Furthermore, a framework *dictates what to be done* during ISD and thus relates to both the target domain and the project domain. A many-to-many relationship exists between ‘framework’ and ‘target domain’ as well as between ‘framework’ and ‘project domain’. Thus, one method chain or method alliance can be used in different frameworks. There is also a many-to-many relationship between method chains and method components as well as between method alliances and method components. A method chain is a *vertical composition* of one or more method components and one method component can be used in different method chains. Similarly, a method alliance is a *horizontal composition* of one or more method components, in which each method component might be used in several method alliances. Finally, a method component is a meaningful assembly of process fragments and product fragments and the concepts used within those fragments. In this context, meaningful refers to appropriateness for addressing a certain aspect of a problem at a potentially re-usable granularity layer.

In relation to the model presented in Figure 2-3, the focus of this work is on the method component part of the model and its relationship to perspective (and hence theory).
2.3.4 Existence Levels of Methods

Goldkuhl (1994) proposes to classify methods according to their level of existence. With that classification methods exist on (1) subjective level, (2) inter-subjective level, (3) linguistic level, (4) action level, or (5) consequence level. The subjective level refers to knowledge and skill held by some individual. The inter-subjective level refers to two or more individuals sharing the same knowledge. The linguistic level refers to externalized prescriptions for action. The action level refers to actions that follow method prescriptions. Finally, the consequence level refers to materialized effects of method-following actions. Two special cases of levels 3 and 4 occur when methods are being implemented in software tools (e.g. CASE tools). Level 3 is then formalized and implemented in software and a computer performs some of the actions of level 4 (possibly in interaction with human method users).

The focus of this work is on methods at the linguistic level, action level, and consequence level; i.e. on levels 3, 4, and 5. Note that this notion of existence levels of methods is expanded upon in Section 3.2 when discussing the more general concept of ‘action knowledge’, of which methods are only one particular form (Goldkuhl, 1999).
2.3.5 Method development

Method development means developing methods. One term that has been used to characterize the structured development of methods is method engineering. Brinkkemper (1996, p. 276) defines method engineering as: ‘the engineering discipline to design, construct and adapt methods, techniques and tools for the development of information systems’. The construction of methods that resides only at a subjective level (in someone’s mind) is thus not regarded as method engineering. In this dissertation method development is considered equivalent to method engineering and, if not explicitly stated, refers to the structured intentional development of methods at the linguistic or action levels (see Section 2.3.4).

Figure 2-4, illustrates the relations between method prescriptions and IS development actions with some important related concepts. The rationale of Figure 2-4 is as follows. Methods direct method users’ attention to certain kinds of phenomena. Some of these are recognized as primitives, i.e. kinds of modelling elements. That is, the focus during method use is determined by the questions asked to abstract analytically. Methods include prescriptions that direct attention towards such primitives. The prescriptions are, in some sense, realizations of the method’s goals (perspective), which are always based on some values. The action prescriptions are then instantiated in actions during method use, which leads to some effect (cf. Section 2.3).

![Figure 2-4: Method as action (Ågerfalk and Åhlgren, 1999).](image)

One way of constructing methods is by reconstructing successful actions (that is actions that lead to desired effects). Method construction can thus be a kind of conceptualization and externalization of already institutionalized implicit methods, in the sense of Berger & Luckmann (1967). Another approach is to begin with some goals or theory and deduce method prescriptions that are consistent with the theory.

The rationale of a method is the argumentation that connects the parts of Figure 2-4 into a congruent whole. With this view method engineering is the process of conceptualizing (creating or assembling) processes, concepts and notations into a congruent whole with respect to some problem domain and desired kinds of solutions (effects). Method engineering also includes the process of externalizing the method in, for example, method handbooks and tool-support (CASE tools).

The argumentation that constitutes method rationale may be built up in three stages (Goldkuhl, 1994). The first stage, internal grounding, concerns how the pieces of a method fit together. That is, an argumentation from a theoretical point of view. This
includes legitimization of method prescriptions by relating them to goals and values. The second stage, external theoretical grounding, relates the method prescriptions to other, external, theoretical knowledge and to other existing methods. The third stage, empirical grounding, concerns empirical tests of the method in use. These activities might very well be performed in parallel with a continuing change of focus between them. Internal arguments are built up partly with influence from existing methods and results from empirical tests, and empirical tests are performed in the light of the theoretical argumentation in a dialectic process. These grounding processes and how they have been carried out in this work are further elaborated in Section 3.2.2.

2.4 Self Criticism

Two relevant questions to ask at this point concerning this study on actability are (a) do we really need yet another ‘-ability’ in the systems development community, and (b) do we really need new methods for specifying and evaluating information systems? The second question can be brought even further and so we can ask ourselves (c) do we really need methods at all? Let us elaborate each of these questions in turn.

2.4.1 Yet Another ‘-ability’?

Certainly, it is legitimate to question if we really need yet another ‘-ability’. We already have ‘us-ability’, ‘port-ability’, ‘learn-ability’, ‘access-ability’ and ‘scal-ability’, to name but a few. This work puts forward a perspective on information systems from which such systems are viewed as tools for business action and communication. The perspective has been made concrete by treating it as one single overall concept: information systems ‘act-ability’. Choosing to propose another ‘-ability’ is due to the fact that the concept embraces the view that information systems have the ability to act, and the ability to support action – information systems have ‘act-ability’, something not usually stressed within other ‘-abilities’.

2.4.2 Yet Another Method?

Whether or not we need another ‘-ability’ (see Section 2.4.1), does the systems development community really need new methods for specifying and evaluating information systems? The question is, however, not fully applicable to this work since no completely new methods are proposed, but re-designs and repackaging of existing approaches. The reason for not settling for what has already been proposed within the systems development field is that existing methods do not seem to give the necessary support for specifying and evaluating information systems in a way that emphasizes the action character of information and information systems needed to design for actability. However, many of the weaknesses in existing methods lie in their perspectives rather than their concepts and notations (see Section 2.3 for a clarification of this distinction). Therefore, many existing notations and modelling formalisms have been retained and re-interpreted in the light of an action view of information systems. The important thing to bear in mind is that the methods proposed are operationalizations of the concept of information systems actability. A concept that has been developed in response to identified weaknesses in contemporary conceptualizations of information systems and information systems use (see Chapter 1). Since we need another perspective, we also, by extension, need further methods that implement it.
2.4.3 Why a Method-Based Approach?

This work proposes a method-based approach to the problem of developing and evaluating information systems. Such an approach can be challenged and there are others. One such alternative approach is to let business actors develop their systems themselves. That approach, often referred to as ‘end-user computing’ (EUC), seems to be a possible alternative with the use of today’s powerful end-user tools such as spreadsheets (Avdic, 1999). An achievement gained with the EUC approach is that the problems of communicating requirements between users and developers are potentially eliminated. With the understanding of the concept of method from Section 2.3, even the EUC-approach can be regarded as a kind of method-based approach that needs support in terms of conceptualizations and forms of documentation.

A potential criticism of methods-based approaches in general is the seeming paradox between the use of a rigorous method (which usually produces lots of documentation) and the need for rapid development of information systems (see Section 8.2). However, informal pre-investigations of this work as well as McConnel (1996), Lam (1998), Martin (1991), and Kruchten (1999) indicates that, if system maintenance and evolution is taken into account, well documented systems and structured engineering practices are imperative to handle changes and to shorten development schedules.

There is also a widespread belief that developing methods and promoting methodical ways of working is doubtful since methods are not used in practice anyway, at least not the way they were intended (Introna and Whitley, 1997; Iivari and Maansaaari, 1998; Nandhakumar and Avison, 1999; Russo and Stolterman, 2000). The position that methods are needed and useful has been challenged and an alternative line of thought ‘Against Method-ism’ (Introna and Whitley, 1997) has been put forth. The argument is that ‘the successful use of methodology [as in ‘method’, see Section 2.3] itself depends on a broader, already present, tacit understanding of the world, an understanding not to be found in any one particular method.’ That is, one single method can never, in itself, compensate for the competent actions of system developers facing real system development situations. As Paul (2002, p. 176) puts it: ‘(There) Is No Substitute For Intelligent Thinking’, a phrase in which ‘There’ can be replaced with anything except ‘Intelligent Thinking’, thus including, for example, ‘A Systems Development Method’ (as well as ‘The Concept of Actability’, for that matter). The thing is that all methods need to be configured to suit the situation at hand (Karlsson et al., 2001; Karlsson, 2002), and they are usually not to be followed slavishly. In this work, the view of method use is that methods should serve as toolboxes (e.g. Nilsson, 1999) rather than as all-encompassing patterns of which particular development efforts are instances. As Goldkuhl (1993, p. 31) puts it: ‘to follow the user manual is not to follow the user manual completely’ (author’s translation from Swedish). That is, method users are considered ‘reflective practitioners’ who are able to choose methods and method parts that suit the particular situation at hand. In order for this to be possible, two things are required. First of all, the practitioner must have something to choose from – one or several methods, or parts thereof, are needed. This does not mean that the methods must be stated explicitly; they may reside solely on the subjective level – inside the method user’s head (see Section 2.3.4). It is likely, though, that an average system developer throughout his or her career has come across one or more externalized methods from which the choice is made, at least partially and/or tacitly. Secondly, the method’s rationale must be communicated (Stolterman, 1991; Ågerfalk and Åhlgren, 1999; Rossi et al., 2000). Method users must be able to understand why particular things are prescribed by the
method in order to make a rational decision. The method rationale – the arguments for
different method prescriptions – is basically what constitutes the method’s perspective
(Ågerfalk and Åhlgren, 1999). In this dissertation, large parts are devoted to this end
specifically – to present reasons as arguments for the proposed methods. These argu-
ments are founded in the concept of information systems actability and its underlying
theoretical basis. Based on an understanding of actability and its operationalization into
method support, method users can make informed decisions about whether to follow
the methods or not. Should no operationalization of the concept be presented, it would
probably be harder for someone believing in it to operationalize it on the fly. This does
not mean, however, that the concept of actability cannot be operationalized in com-
pletely different ways, or that it cannot be integrated into other methods at the level of
perspective, but that is outside the scope of this dissertation (see Section 16.4.2 for
possible future research in this direction).
Chapter 3

Research Design

The main purpose of a research design is to help avoiding the situation in which the evidence does not address the research questions (Yin, 1994). According to Yin (1994), five components of a research design are especially crucial:

1. A study’s questions.
2. Its propositions, if any.
3. Its unit(s) of analysis.
4. The logic linking empirical observations to the propositions, and the criteria for interpreting the findings.

In Chapter 1, the overall question for this dissertation was described in terms of how information systems usage can be understood from a social action perspective and what consequences such an understanding has for the design of information systems, and specifically how such an understanding can contribute to the design of information systems so that they will be more satisfactory.

Supported by, for example, Hirschheim et al. (1996), the main proposition, or thesis, upon which this dissertation rests, is that information systems design can be improved if information systems are understood from a social action-perspective. This was described in Chapter 1 as a reconciliation of various views on usage quality into the concept of IS actability. Chapter 1 also introduced three different operationalizations of actability: a method for developing information systems, a method for evaluating information systems and an analytic framework for understanding IS related phenomena, summarized under the headings of Actability Design, Actability Evaluation and Actability Studies, respectively. These three operationalizations can be understood as the three main units of analysis, alongside the concept of IS actability itself.

In the remainder of this chapter, these are considered in more detail and a conceptual link is established between them, other theoretical knowledge, and the empirical findings. In doing so, some ontological and epistemological foundations for the dissertation are touched upon, starting with an adopted framework for in-context interpretative research. We then move on to the concepts of knowledge and grounding of knowledge.

3.1 A Setting for Interpretative in-Context Research

The research approach adopted can be characterized as an interpretative in-context research approach. Interpretative research is often contrasted with the positivistic research tradition (e.g. Walsham, 1993; 1995a; Braa and Vidgen, 1999). While positivists reckon with an objective reality, interpretivists acknowledge that different people will interpret a situation differently. ‘Typically, positivism is concerned with reducing the area of investigation in order to be able to make reliable predictions and explanations, while interpretivism is concerned with making a reading of a situation in order to gain understanding.’ (Braa and Vidgen, 1999, p. 27)

When acting, such as doing empirical research, one intervenes in the world. Even if a researcher is acting as an external observer of the world, his or her mere presence
makes the situation different compared with his or her absence, and hence he or she intervenes (Braa and Vidgen, 1999). In action research, the intervention is deliberate and part of the research goal is to bring about change to a problematic situation (Mumford, 2001).

Based on these categories, three general overall goals with research can be identified as bringing about change, making predictions, and gaining understanding (Braa and Vidgen, 1999). According to Braa and Vidgen (1999), these three goals are not mutually exclusive, but rather all three are present to a varying extent in any research setting. To show how they relate they introduce the model depicted in Figure 3-1.

![Figure 3-1: Information systems research as interpretation, intervention and reduction (Braa and Vidgen, 1999).](image)

One strength of this model is that it makes it possible to position a particular research design along the scales of reduction, intervention and interpretation, and consequently positioning the research aim with respect to its bias towards prediction, change and understanding.

As we shall see in Section 3.2, the research approach adopted in this work includes both empirical action research, such as the practical application of specifying and evaluating information systems in the field, as well as reflective interpretative analyses (Alvesson and Sköldberg, 2000; Walsham, 1995b) of commonly recognized and understood cases, such as the case of the Automatic Teller Machine elaborated in Chapter 7 and the car sales process elaborated in Chapter 6. This means that the research reported in this dissertation is biased towards understanding and change applying a combination of interpretative and intervening research strategies.

This position does not imply that making predictions is unimportant. In fact, much of the work reported is expressed in the form of method support for systems work (including design and specification as well as evaluation). As discussed in Section 2.3,

---

8 In this work, the phrase ‘action research’ is used to mean research that is deliberately oriented towards intervention (and thus change, see Figure 3-1): ‘research that involves practical problem solving which has theoretical relevance’ (Mumford, 2001, p. 12). This seems to be in line with the common use of the phrase within the IS community (see Avison et al., 1999; McKay and Marshall, 2001).
Information systems actability

methods represent prescriptive normative knowledge and are, as such, based on an underlying assumption that following the method will eventually lead to desired results (Russo and Stolterman, 2000). In that sense, methods, as a form of knowledge, represent predictive knowledge. The thing is that this work does not generally seek predictive knowledge in terms of reductionistic universal laws but in contextual pragmatic guidelines applicable in practice. This goes back to the design orientation inherent in informatics research in general (see Section 2.1) and to the fact that our main interest as informatics researchers is ‘in the contingent and exceptional more than in the general, in local design principles more than in general laws, … in the good and beautiful more than in the true’ (Dahlbom, 1996, pp. 42-43).

3.2 Conceptualization and Application of Knowledge

The concept of IS actability can be regarded as an instance of what Goldkuhl (1999) refers to as action knowledge, that is, ‘theories, strategies and methods governing people’s action in social practices’ – it represents knowledge about action, and it is intended for action. In fact, it is arguable that action is actually where the value of knowledge is ultimately achieved; of what value would knowledge be if we did not make use of it to act in the world? (Cf. Argyris et al. (1985) on the distinction between knowledge in the service of action and knowledge for its own sake.) Therefore, the two terms ‘knowledge’ and ‘action knowledge’ are used interchangeably in the remainder of this dissertation.

3.2.1 Ontological Foundation

All interpretative research relies fundamentally on the ontological standpoint of the individual researcher (Walsham, 1993). The ontological foundation on which this research is founded is based on what Goldkuhl (2002) terms socio-instrumental pragmatism. The term reflects an emphasis on humans acting in a social world, supported by various instruments (such as language and other external objects). Figure 3-2 depicts the main categories within the ontology.

Inspired by the three worlds ontology by Popper (1975) and Habermas (1984), the point-of-departure for the ontology adopted in this research is humans acting in an external world to the actor by intervening in it and thus changing it, and interpreting it in attempts to make an understanding of it. The external world thus consists of various artefacts (such as IT-systems) and communicative signs (such as messages) as results of action. Humans act in a natural environment, and the external world thus consists also of different natural phenomena (such as waterfalls and the sun). Acting in the world requires an understanding of it, acquired through earlier interpretations. That subjective understanding, together with the person’s intentions, abilities, emotions, etcetera can be referred to as a human’s personal cognitive base. The personal cognitive base, as well as the person themselves is part of that person’s subjective world. In order for two people to communicate, they must share a common understanding of the world communicated about as well as of how to communicate about it. Such common understanding can be referred to as the humans’ shared cognitive base, which together with the people involved constitute their inter-subjective world. In addition to performing interventions directed towards the external world, one can perform reflective action based on and directed towards one’s own subjective world.
In Chapter 4 this ontology is expanded to form a basis for conceptualizing information systems and their business use. In order to ground the research approach, let us now focus particularly on its relationship to knowledge.

![Diagram](image)

**Figure 3-2:** Different realms of the world (adapted from Goldkuhl, 2002).

In the spirit of the proposed ontology, Goldkuhl (1999) claims that action knowledge may exist on five different levels: a *subjective* level, an *inter-subjective* level, a *linguistic* level, an *action* level and a *consequence* level. Knowledge at a subjective level is part of that human’s personal cognitive base and is related to the notion of ‘tacit knowledge’ (Polanyi, 1958). Knowledge at an inter-subjective level is knowledge shared by several people as parts of their inter-subjective worlds. The linguistic level refers to knowledge that is expressed as communicative signs, for example, as written strategies or policies. As the name suggests, action knowledge is expressed, or manifested, in action. This is the action level of knowledge. Finally, traces of the action knowledge might be found in materialized artefacts, which constitute the consequence level. Since what we refer to as one piece of knowledge may exist simultaneously at several of these different levels, it seems more adequate to speak of the ‘levels’ as different aspects of knowledge. Let us consider an example to clarify these concepts. In order to study the construction of a particular IT-system, we can choose to investigate the actual system, which would constitute an artefact existing in the external world. To get more insight, we may also consult different descriptions and models of the system produced during the development process. These would then constitute communicative signs created during the development process as a means to get inter-subjective understanding of the system. Another way to learn about the system would be to study developers’ actual performance of development actions, which would be governed by the developers’ subjective and inter-subjective understanding of the system to be built as well as of systems development in general. From a research point-of-view, it is important to see that tacit as well as non-expressed linguistic knowledge are *unobservable*
while explicitly stated linguistic knowledge as well as actions and consequences are observable.

It is furthermore important to realize that knowledge that is viewed as a consequence in one context might be viewed as a tacit aspect, linguistic aspect or action aspect in another, depending on what we choose to focus. That is, a piece of knowledge must always be understood in relation to other pieces of knowledge. This recursive nature of action knowledge is returned to below when the grounding of such knowledge is discussed. The empirical work on actability is mainly concerned with inter-subjective observable knowledge, such as method prescriptions with method-following actions and consequences. Nonetheless, for the concept of actability, unobservable knowledge is important since action is always influenced by unarticulated prerequisites, which of course complicate the design of information systems.

3.2.2 Principles for Grounding of Knowledge

The proposed ontology underlying this work (Section 3.2.1) has consequences also for the generation and validation (grounding) of knowledge – that is, it has epistemological consequences. In order for a piece of knowledge to make sense to someone it must be possible to integrate with that person’s personal cognitive base – it must be possible to include in that person’s subjective world. Such internalization means to justify the knowledge for oneself – to rationalize it. To do that, the person may compare the knowledge with previous experiences internally (intrapersonally); matching it with other people’s beliefs and intentions by way of shared cognitive bases; and trying it out in the external world to get empirical justifications.

This rationalization of new knowledge may be compared with Weber’s (1978) notion of practical rationality. According to Goldkuhl (1999), such a notion is necessary in order to ground action knowledge since prescribed means must be possible to relate not only to empirically observable consequences but also to intended ends and values. According to Weber (1978), rationality consists of three sub-rationalities: instrumental rationality (means in relation to ends), rationality of choice (ends in relation to values) and normative rationality (ethical principles in relation to action). Furthermore, the concept of grounding of action knowledge assumes that rationality is argumentative and discursive, which means that it is possible to argue the validity of the knowledge in inter-subjective dialogues (Goldkuhl, 1994). This view, with its roots in Habermas’ (1984) social-critical concept of rationality, is the key to an important distinction between true and valid – something is ‘true’ for somebody if that somebody accept it as valid and useful. What is ‘true’ is that which is socially agreed upon. ‘Claiming the validity of knowledge is presenting good reasons as arguments for the knowledge.’ (Goldkuhl, 1999) Hence, grounding of action knowledge is to present such good reasons for it that other people accept it as valid. This is supposed to be done by argumentatively relating the focused knowledge to three different knowledge sources acting as warrants for the knowledge under scrutiny. These three knowledge sources give rise to three different grounding processes. Understanding these three grounding processes is important in order to understand how the validity of actability can be claimed.

First, the knowledge can be related to its own background knowledge and hence knowledge partly holds its own justification. This internal grounding means to reconstruct and articulate knowledge that might be taken for granted. All explicated knowledge can be understood in terms of the concepts used to articulate the knowledge. To analyse how these concepts relate, and to define the concepts as such, is to perform a
conceptual grounding of the knowledge. Another important aspect of the articulation of the background knowledge is to relate it to its values and goals. Furthermore, the concepts used and their anchoring in values need to be consistent and free from ambiguities and internal contradictions. This can be achieved through an explicit evaluation of the cohesiveness of the knowledge. Second, the knowledge can be related to other existing knowledge of theoretical character. This *external theoretical grounding* means to perform conceptual grounding and value grounding where relations between the focused knowledge and other external knowledge are in focus. It might also mean the grounding of the knowledge in existing explanatory theories. Third, the knowledge can be related to empirical observations. This *empirical grounding* means to ground the knowledge through applications and observations. The three grounding processes are summarized in Table 3-1.

| **Table 3-1**: A summary of the different grounding processes (Goldkuhl, 1999). |
|---------------------------------|---------------------------------|---------------------------------|
| **Internal grounding**         | **External theoretical grounding** | **Empirical grounding**        |
| ê Knowledge reconstruction     | ê Conceptual grounding           | ê Application and observation grounding |
| ê Conceptual grounding         | ê Value grounding                |                                 |
| ê Value grounding              | ê Explanatory grounding          |                                 |
| ê Evaluation of knowledge cohes ion |                                 |                                 |

### 3.2.3 Concepts, Operationalizations, Applications and Consequences

In addition to the above mentioned aspects of knowledge existence, action knowledge might exist in different forms of abstraction – from ‘pure’ abstract theoretical knowledge to knowledge directly applicable in everyday situations. Since the concept of actability is an abstract concept, there has been a need to operationalize it into more applicable and empirically testable forms. The idea of operationalization assumes two basic categories: the *concept* and the *operationalization* of a concept. The concept is always more abstract than its operationalization and both need to be an explicitly stated linguistic aspect of the action knowledge under scrutiny. This is necessary in order to observe and communicate the analysis between researchers and reporting the results of the research. Of course, they need not be explicated initially, even though it is preferable since the externalization of knowledge into written formulations requires precision and hence the very externalization process becomes an important part of internal grounding of both the concept and its operationalization. Thus, the formulation and externalization of a concept and its operationalization implies an internal and external theoretical grounding. The operationalized concept can then be applied in practice whereupon consequences arise. Let us refer to these two instances of the action knowledge as *application* and *consequence*, respectively. The first represents an action aspect of the knowledge and the second represents a consequence aspect. Both instances are possible to relate to empirical observations and empirical data and thus together constitute the empirical grounding of the action knowledge. To summarize, one piece of knowledge can be instantiated and studied in (at least) four different shapes: as a concept, as an operationalization of the concept, as an application of the operationalization, and as a consequence of the application. Table 3-2 concludes this discussion with actability as the main concept for this work with its three operationalizations.
Note that the consequence instance of the action knowledge, as well as an operationalization, can be considered as a concept in its own right. That is, we can choose to focus, for example, the consequence as ‘the piece of knowledge’ and operationalize it, *et cetera*, to eventually arrive at even more specific applications and consequences. As suggested above, this recursive character is inherent in the nature of action knowledge. In the practical research work, this recursiveness has been used as a means to go further into specific applications (see Section 3.4, specifically Table 3-3, p. 46). It has also been used for the purpose of grounding actability *per se* as described below.

**Table 3-2:** Operationalizations of actability with their applications and consequences.

<table>
<thead>
<tr>
<th>Concept</th>
<th>Operationalization</th>
<th>Application</th>
<th>Consequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actability</td>
<td>VIBA</td>
<td>Specification of requirements</td>
<td>Requirements specifications</td>
</tr>
<tr>
<td>Actability</td>
<td>Evaluation Method</td>
<td>Evaluation of existing and re-designed systems</td>
<td>Evaluation reports and re-design suggestions</td>
</tr>
<tr>
<td>Actability</td>
<td>Analytic Framework</td>
<td>Analysis of IS phenomena</td>
<td>Description of IS phenomena</td>
</tr>
<tr>
<td><strong>Internal grounding and</strong></td>
<td></td>
<td><strong>Empirical grounding</strong></td>
<td></td>
</tr>
<tr>
<td><strong>External theoretical grounding</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 3.3 Research Strategy

Within the concept of grounding of knowledge, we must conceptually distinguish between the *generation* and the *validation* of knowledge. Even though the two are usually quite intertwined, it is possible to talk of a ‘context of discovery’ and a ‘context of justification’ (Goldkuhl, 1999). Traditionally, knowledge is generated and validated either inductively or deductively. In the case of induction the knowledge source for generation is purely empirical observation. Some qualitative researchers, such as early advocates of Grounded Theory (e.g. Glaser and Strauss, 1967), even claim that knowledge should be strictly generated from empirical data and not from preconceptions and pre-formulated hypotheses. With a deductive approach knowledge is generated from external knowledge of theoretical character or simply through ‘internal’ construction of hypotheses. The knowledge is then validated through, for example, empirical observations.

An alternative approach to strict induction or deduction is what can be referred to as a reflective approach (Alvesson and Sköldberg, 2000), which also seems to be acknowledged in more recent approaches to Grounded Theory (e.g. Strauss and Corbin, 1998). The idea is to continuously develop theory based on external theories and on empirical observations and to let the evolving theory play an active part in collecting and interpreting data in a recursive manner (cf. Walsham, 1993). One reason for adopting such an approach is that it is impossible to be completely free from bias (Strauss and Corbin, 1998) and being explicit about preconceptions makes the research process more transparent, and the results more valid (cf. Yin, 1994). A reflective approach helps to remedy some of the problems with a strict inductive approach, as that it helps direct attention to relevant phenomena. It is also a means to develop knowledge that is known to be practically relevant. Reflective research also means that interpretation
becomes central. Interpretation takes place when a researcher works with theory and data, theory is interpreted and related to empirical data, and empirical data is interpreted and related to evolving as well as externally existing theory. A reflective approach also means that the research process per se is considered a subject for constant interpretation and refinement (Alvesson and Sköldberg, 2000). In a sense, such a research strategy implies aiming at several moving targets simultaneously. To handle that complexity, it is important that researchers are sensitive towards their own actions and conceptions; that is, reflect upon what they are doing.

In this research, a reflective approach has been adopted partly because interpretation is always a part of research, just as it is a fundamental part of life in general. This epistemological standpoint implies that not recognizing research as an interpretative process is fundamentally ignorant, and has therefore been rejected. By acknowledging humans as subjects with free will trying to make sense of the world by interpreting it based on their own preconceptions, the fallacy of regarding scientific results as objective truths can be avoided. As stated above, from a pragmatic standpoint, the issue is not whether something is true or not, but whether it is useful in practice. In order for a researcher to build up his confidence in the results, which is a necessity if others are to be convinced, first hand experience of the phenomena of study is imperative. This idea goes well with a reflective approach since data, abstractions, and theory, as well as the very process of working with these phenomena can be continuously evaluated, refined and expressed.

In interpretive research, theory plays an important role in that an interpretation is always based on the interpreter’s background knowledge. In this work, theory and conceptions of information systems and their use within businesses has been the object of study. Therefore, theory has had to play a central role; not only as background knowledge, but also as a primary phenomenon under investigation. To neglect initially evolving theory in subsequent research efforts would be like buying a bicycle and continuing to undertake the one-hour walk to the office every morning, without reflecting over the fact that using the bike would not only be more time-efficient, but also more convenient. However, to take the bicycle everyday not taking into account that on rainy days walking would make using an umbrella possible would be equally unreflective. Not questioning this, if going to work was necessary, would be even worse. In the research on IS actability there has certainly been a need to question whether to go to work or not, that is, to continuously refine the theory and to reflect over its pros and cons, and also to admit that sometimes the wrong road was chosen. In such cases, neither walking nor cycling would make things better, one cannot compensate wrong direction with speed.

The empirical research on actability, as reported in this dissertation, has been conducted according to a case research strategy. Yin (1994, p. 13) defines ‘case study’ as ‘an empirical inquiry that investigates a contemporary phenomenon within its real-life context, especially when the boundaries between phenomenon and context are not clearly evident’. He further argues that a case study approach is most beneficial when the study’s questions concerns ‘how’ and ‘why’ questions.

There is no question that the borders between phenomenon and context have been hard to distinguish in this research. This follows naturally from the intimate relations between concepts, operationalizations, applications and consequences, as described above (Section 3.2.3). From the outset, it was unclear which were the units of analysis (i.e. the phenomena to study). Consequently, the borderline between phenomenon and context has been allowed to shift back and forth over time in a dialectic manner. Some-
times the concept has been in focus, with its operationalization as background. Sometimes it has been exactly the other way around. Such focus shifts are essential in order to succeed with the chosen approach that intertwines the context of discovery and context of justification (see above).

This research concerns how to understand information systems from a social action perspective. In order to explore these ideas, digging deeper into a few cases has been regarded as the most rewarding approach. According to Yin (1994), the unit of analysis is related to the fundamental problem of describing what the ‘case’ is. ‘The key issue in selecting and making decisions about the appropriate unit of analysis is to decide what it is you want to be able to say something about at the end of the study.’ (Patton, 1990, p. 168) In this research, the high level units of analysis have been the three operationalizations of actability, and by extension the concept of actability per se. The reason for treating the operationalizations as units of analysis is that the operationalizations have evolved over time and there has been much to gain from tracking this evolution.

The research design should be regarded as an embedded one (Yin, 1994), which means that different levels of empirical units have been studied. As we shall see later in the dissertation, for example, the work on Actability Design has involved several different settings with different foci ranging from professional developers working with the proposed method to pure theoretical reflection. Figure 3-3 summarizes the overall units of analysis used in this work.

As depicted in Figure 3-3, the different units of analysis can be allocated to three distinct levels (or focal areas). These are referred to as the levels of theory, practice and approach. At the practice level, the research is concerned with actors who, within a business context, perform business actions through and by means of information technological artefacts. These three components, together referred to as the A³ model (see Figure 1-4, p. 11), constitute different units of analysis, which must be understood in relation to each other; for each unit the other two constitute its context. Different practices have been studied by use of the evolving actability operationalizations. These operationalizations, per se, constitute further units of analysis. For the aim of this dissertation, the interest is in different properties (constituents) of the operationalizations and to what extent they direct attention to, and help to explain, relevant phenomena within the practices studied. This includes the attitudes of users of the operationalization, such as researchers and other external participants, and how the understanding of the practice and the operationalization develops. At the level of theory, this research is concerned with the concept of actability and the repercussion the development of knowledge of the other two levels has on it.

If we compare Figure 3-3 with Table 3-2 and the discussion in Section 3.2.3, we can see that the level of theory constitutes the overall concept, which is operationalized into the level of approach. The operationalization is then applied to study the level of practice. During such applications, consequences arise. All three levels contain units of analysis, which when focused constitute concepts in their own right.

The case research strategy is often associated with qualitative research approaches (e.g. Benbasat et al., 1987). In line with that notion, the research on actability has been performed according to a qualitative research tradition. The reasons for this are derived from the aims of the studies. According to Patton (1990, p. 167): ‘One of the strengths

---

9 The cases upon which this dissertation is based are referred to as Applications 0–7, and Study I and II. These are described in detail in chapters Chapters 8 (Applications 1–4), 11 (Applications 5–7), 14 (Study I) and 15 (Study II).
of qualitative analysis is looking at program units holistically. This means doing more than aggregating data from individuals to get overall program results. When a program, group, or community is the unit of analysis, qualitative methods involve observations and descriptions focused directly on that unit. Hence, a qualitative approach offers support for the required focus shifting between the different embedded units of analysis at different levels, as depicted in Figure 3-3.

Figure 3-3: Principal units of analysis.

The initial research\(^\text{10}\) was an explorative study in which early ideas were tried and refined. As the concept of actability stabilized, the research approach to use was a more open question. The reason for favouring qualitative action research was inherent in the research problem as such. We wanted to gain experience, and draw conclusions, from working with different operationalizations of actability. The operationalizations are complex phenomena \textit{per se}. In addition, they were under development and continuously change. Such circumstances require closeness between the observer and the observed in order to gain rich and balanced data. Closeness in itself does not necessarily imply action research; to that end observations would suffice. However, in accordance with the reflective stand taken in this work, active participation makes it possible to

\(^{10}\) Note that we will not go into details of the three actability operationalizations at this point. Instead, the research method adopted in each case, as well as the selection of cases, will be described together with the operationalizations (Chapters 8, 11, 14 and 15).
adapt the study to changing circumstances and, as has been central to this work, try out emerging ideas instantly. Furthermore, the circumstances made it difficult to specify expected results in advance, which also speaks in favour of a qualitative approach.

### 3.4 The Grounding of Actability

The grounding of actability follows what has been discussed above (Section 3.2.2), i.e. internal grounding, external theoretical grounding and empirical grounding. Note that grounding refers to both generation and validation as two important aspects knowledge development. The main sources for external theoretical grounding has been various contemporary IS knowledge domains, primarily information systems development (ISD) and requirements engineering (RE) (e.g. Iivari and Lyytinen, 1998; Hirschheim et al., 1996; Sommerville, 1996; Graham, 1998) and communication modelling, including the language/action perspective (e.g. Winograd, 1988) and organizational semiotics (e.g. Stamper, 2001). Additionally, usability-related knowledge (e.g. Norman, 1988; Bevan, 1995a; 1999; Carroll, 1997) and also to some extent Activity Theory (AT) (Kuutti, 1996) and Actor Network Theory (ANT) (Latour, 1991; Monteiro, 1999) have influenced the work. The different external sources of knowledge have been reconciled and re-interpreted from a social action perspective to form the concept of actability – the internal grounding process.

As mentioned above, actability has been operationalized in three different ways for the purpose of empirical grounding (see Table 3-3). Note that in Table 3-3, the operationalizations are labelled concepts, as distinct from Table 3-2 in which they were referred to as operationalizations of the concept of actability. This is due to the recursive nature of action knowledge discussed in Section 3.2.3.

<table>
<thead>
<tr>
<th>Concept</th>
<th>Internal Grounding</th>
<th>External Theoretical Grounding (Main Sources)</th>
<th>Empirical Grounding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actability</td>
<td>Purely internal</td>
<td>ISD + RE</td>
<td>An ISD method (VIBA)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CM (LAP + OS)</td>
<td>An actability evaluation method</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Usability</td>
<td>An actability analytic framework</td>
</tr>
<tr>
<td>VIBA</td>
<td>Internal and in relation to actability</td>
<td>RUP/UML</td>
<td>Development projects</td>
</tr>
<tr>
<td></td>
<td></td>
<td>User Centred Design</td>
<td>Classroom experiences</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Rapid Application Development</td>
<td></td>
</tr>
<tr>
<td>Evaluation Method</td>
<td>Internal and in relation to actability</td>
<td>Usability evaluation methods</td>
<td>IS evaluation projects</td>
</tr>
<tr>
<td></td>
<td></td>
<td>General evaluation methods</td>
<td></td>
</tr>
<tr>
<td>Analytic Framework</td>
<td>Internal and in relation to actability</td>
<td>The semiotic framework</td>
<td>Analysis and description of the Internet-based information system</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The notion of practice</td>
<td>Analysis and description of the local electronic marketplace</td>
</tr>
</tbody>
</table>

Table 3-3: Grounding of the different concepts.
The first operationalization, a systems development method referred to as VIBA. The second operationalization is in the form of a method for evaluation of actability. The third operationalization is in the form of an analytic framework intended for analysing and understanding information systems phenomena. Since action knowledge is recursive by nature, so is the process of grounding it. Therefore, a particular concept (see above) can be grounded both internally, externally and empirically. The result of (or rather the operationalization used for) empirical grounding can then be focused as a concept in its own rights to which the three grounding processes apply as well. Table 3-3 shows the three groundings of actability per se and of its three operationalizations. The adopted principle is that the grounding of an operationalization, for example the actability evaluation method, internally, externally and empirically yields an empirical grounding of actability, which is the main concept under scrutiny. As shown in Table 3-3, the internal grounding of an operationalization includes the grounding of it in its actability ‘legacy’ as well. As introduced above, actability has been externally grounded mainly with respect to ISD and RE, CM and usability. Each operationalization has then been externally grounded in more specific approaches within these larger areas, as shown in Table 3-3. In the next section, the grounding of these operationalizations is presented.

3.5 The Research Process

A reflective approach that acknowledges all three sources of knowledge (internal, external and empirical) as important for both generation and validation has been used in this research. With that approach, we can use different external theories as a source for generating knowledge. This knowledge can then be applied in practice and empirical data collected, which typically will have repercussion on the knowledge under development. The knowledge can also be related to other external theories (and/or the ones used for its generation) to achieve external theoretical validation. This way generation and validation go hand in hand during an accumulative grounded knowledge development.

The research on actability has been going on since early autumn 1997 when the redesign of VIBA was initiated. The research process, as described above, implies that the internal grounding and external theoretical grounding of actability has been highly intertwined with its empirical grounding and hence with its different operationalizations. This has called for two generic foci, referred to as conceptualization (internal grounding and external theoretical grounding) and application (empirical grounding).

The empirical research within this overall framework has been carried out differently within different empirical contexts. The all-pervading approach, however, is the reflective approach introduced above.

Within this framework, an operationalization of actability is applied in practice and data continuously collected, typically through interviews, observations and field notes based on experiences from interventions (i.e. action research). The data collected is then analysed and suggestions for modification of the operationalization identified. As a means for structuring the analysis, the data is categorized and interrelated in a structured way, similar to that of Grounded Theory (Strauss and Corbin, 1998). To this end, conceptual modelling (typically entity-relationship modelling with UML-like notation) is used as a way to picture and describe the generated categories. During analysis, act-
ability and the focused operationalization are used as tools that direct attention to phenomena relevant for the situation at hand. During this iterative process of operationalization, application and conceptualization, repercussion on the concept of actability itself occurs as a natural consequence. This way actability is continuously developed conceptually while it is being empirically validated.

Exactly how the approach presented has been used for the three different operationalizations will be elaborated in Chapters 8, 11, 14 and 15 in conjunction with detailed descriptions of the three actability operationalizations.

### 3.6 A Note on Relevance, Reliability and Validity

When judging the quality of research it is common to apply the quality criteria relevance, reliability and validity.

The question of a study’s relevance may be approached from two perspectives. A study may be scientifically relevant but have limited practical usefulness. On the other hand, a study may have high practical value but suffer from limited theoretical contributions. To some extent, the proposed view on research as constructing well-grounded arguments for what is usable rather than for what is true means that relevance becomes an integrated part of validity. Something that is irrelevant can simply not be useful. Consequently, in the work on actability, relevance has been held as a high ideal. The all-pervading approach to achieve practical relevance has been to work closely together with practitioners, both developers and users. To ensure scientific relevance, measures have been taken to continuously expose the evolving results to different scientific communities, for example, through participation at conferences and workshops on various subjects, as well as more informal seminars with different research groups.

Reliability has to do with whether a study would yield the same results given that it was performed once again, perhaps by another researcher. Generally, it is a bit problematic to talk about reliability in relation to qualitative studies. This is rooted in the fact that interpretation by definition makes it impossible to guarantee reliability in the traditional sense. Therefore, it is more important that a study is argumentatively valid. If the underlying conceptualizations and choices made are transparent, the study is reliable. Therefore, the author strives to be explicit on these points throughout the dissertation, and also present how the concepts have evolved over time.

A related issue is the question of the possibility of drawing general conclusions from only one or a few cases, which is inherent in the research design adopted in this work. According to Walsham (1993, p. 15), ‘the validity of an extrapolation from an individual case or cases depends not on the representativeness of such cases in a statistical sense, but on the plausibility and cogency of the logical reasoning used in describing the results from the cases, and in drawing conclusions from them’. This is in line with Yin (1994) who argues in favour of analytical abstraction rather than statistical. That is, ‘cases are not “sampling units” and should not be chosen for this reason. Rather, individual case studies are to be selected as a laboratory investigator selects the topic of a new experiment… the method of generalization is “analytic abstraction”, in which a previously developed theory is used as a template with which to compare the empirical results of the case study’ (Yin, 1994, p. 31). Again, it seems to be a question of argumentative rationality, of providing good reasons as arguments for the claimed knowledge. After all, ‘There are no perfect evaluation designs, only more and less useful ones.’ (Patton, 1990, p. 168)
3.7 The Validity of Actability

This chapter has presented the process of grounding (generating and validating) the concept of IS actability. Three generic grounding processes with corresponding knowledge sources have been discussed: internal grounding, external theoretical grounding and empirical grounding, and specifically how these have been used to ground the concept of IS actability. The research process has been described as a reflective, case study based qualitative approach where the actability concept has been generated and validated by means of three, partly overlapping, operationalizations. The first focused on actability as a foundation for information systems development, the second on evaluation of actability in existing systems, and the third on using actability as a means to understand information systems phenomena. Now it is time to conclude. Is the concept of actability valid, and if so, in what sense and to what extent? The very concept of validity, as discussed in this chapter, implies two contradictory conditions: (1) all knowledge is valid and (2) no knowledge is valid. The validity of actability has been argued by means of a presentation of the different grounding processes and activities performed to that end. Now it is up to you as a reader to pass the sentence. If you (after considering the rest of the dissertation, of course) believe that what has been proposed means that actability is valid, then so it is. If you agree that this work contributes to the understanding of information systems as tools for business action and communication, then so it does.
Part II

The Concept of Information Systems Actability
Chapter 4

A Pragmatic View on Information Systems


4.1 Chapter Outline

This chapter builds on the ontology introduced in Section 3.2.1 and presents a general model of social action, which is related to the so-called semiotic ladder (Stamper, 1996; 2001). The model is then used to characterize information systems as artefacts used for organizational action. Building on that characterization, information systems usage in terms of supporting human action is analysed within the concept of actability. Finally, some implications for the design of information systems are discussed. The main message of the chapter is that information systems should be conceived as, and consequently designed to be, systems intended for action. The chapter is structured as follows. After introducing some basic concepts in Section 4.2, different interpretations of information systems pragmatics starting from the notion of the semiotic ladder are discussed in Section 4.3. In Section 4.4 information systems are related to concepts of action, organization and artefact. A generic action model is presented as a basis for the actability concept. The key concept of this dissertation, information systems actability, is thoroughly discussed in Section 4.5 and some implications for information systems design are presented in Section 4.6. Finally, intermediate conclusions are presented in Section 4.7.

4.2 Setting the Scene

The many failures of IT-based information systems give rise to a strong imperative for researchers in informatics and neighbouring disciplines to create a better understanding of the nature of such systems and their organizational use (Orlikowski and Iaccono, 2001). As discussed in Chapter 1, just viewing an information system (IS) as a technical black box having some social and organizational effects is not enough. We must understand information systems in a deeper sense than as just one kind of technical artefact. What special kind of artefact is an information system?

A semiotic and communicative perspective seems to offer great possibilities for such a deeper understanding (Stamper, 1996; 2001; Andersen, 2001). Starting from the notion of sign, we can focus on different aspects of information systems as sign systems, ranging from purely technical to social and organizational issues. Based on the work by Morris, three levels (or aspects) are distinguished in classical semiotics: syntactics, semantics and pragmatics. Stamper (1996; 2001) adds some more levels and
Information systems actability

constructs a ‘semiotic ladder’, with the following steps from bottom to top: physical world, empirics, syntactics, semantics, pragmatics and social world.

The pragmatic dimension of human communication has been studied and conceptualized within the theories of speech acts (Austin, 1962; Searle, 1969) and communicative action (Habermas, 1984). A ‘language/action’ perspective on information systems has been articulated by several scholars, taking their main inspiration from these theories (e.g. Goldkuhl and Lyytinen, 1982; Winograd, 1988; Dietz, 1994). The fundamental speech act thesis is that communication is to be seen as one kind of action; that an utterance (message) is a combination of a propositional (informational) content and an illocutionary force (an action mode). The propositional content is what is talked about and the illocutionary force means what kind of action is performed (Searle, 1969). The illocution used is a result of the intention behind the communication. When communicating, we formulate propositional contents and embed these in communication action types. In an attempt to systematize this notion of a communication act, Searle (1969) made a distinction between four different sub-acts of a speech act:

Â the utterance act,
Â the propositional act,
Â the illocutionary act, and
Â the perlocutionary act.

The utterance act is to be seen as the production of a sequence of words that together form a comprehensible whole. This can be understood mainly as equivalent to the syntactic level. The propositional act means referring and predicating, i.e. representing a world talked about in an utterance. This corresponds to the semantic level. The illocutionary act is what we are doing by speaking, for example, stating, commanding, promising or declaring. The perlocutionary act is the intentional ‘causing’ of effects in listeners. The relationships between these two last sub-acts and the semiotic aspects are not straightforward and are further elaborated in Section 4.3.

To distinguish between utterance, proposition, illocution and perlocution seems to be important. Nonetheless, it may be misleading to describe these as different sub-acts performed within a speech act. They are rather different aspects of a speech act. These notions are introduced here, as they will be used later in the chapter.

A pragmatic perspective on information systems seems to offer good possibilities for a deeper understanding of such systems as artefacts in organizational settings. The purpose of this chapter is to explore the pragmatic character of information systems to arrive at such an understanding. Taking the semiotic ladder as the point of departure we eventually arrive at the concept of information systems actability, defined as an information system’s ability to perform actions, and to permit, promote and facilitate the performance of actions by users, both through the system and based on information from the system, in some business context. In conducting the journey, we investigate possible meanings of pragmatics related to information systems and touch upon some implications for information systems design that are returned to later in the dissertation.

4.3 Information Systems and Pragmatics

Stamper (1996; 2001) presents a semiotic ladder (the semiotic framework) for analysing different aspects of information systems as sign systems. This organizational semiotics approach to information systems has had a substantial influence on the IS field
A pragmatic view on information systems

and several authors have both used and contributed to it (see, e.g. Liu et al., 2001a). The semiotic ladder consists of the following steps: physical world, empirics, syntactics, semantics, pragmatics and social world. The pragmatic level is defined in the following way (Stamper, 1996): ‘Pragmatics is the branch of semiotics concerned with the relationships between signs (as meaningful utterances) and the behaviour of responsible agents, in a social context.’

The relationship between action (i.e. ‘behaviour of responsible agents’) and signs is, however, not unambiguous. It is possible to identify different relationships. Figure 4-1 illustrates this reasoning.

![Diagram of message production]

**Figure 4-1:** Different kinds of actions related to signs (Goldkuhl and Ågerfalk, 2002).

A message (depicted in Figure 4-1) can be produced by a human agent or an IS. This message (a sign constellation) is received and interpreted by some human agent. After interpreting the message, the agent performs some action based on that interpretation. The common understanding of the pragmatic aspect of information systems seems to be this kind of ‘ex post action’, i.e. an action based on messages from an IS (e.g. Langefors, 1995). The pragmatic aspect of information systems is concerned with the actions performed by humans after receiving messages from information systems. In what ways are humans influenced by messages from information systems? What are humans doing when using messages from information systems as a base? This is one interpretation of pragmatics, but not the only one.

Let us look more closely at the message, which can be produced by an IS or a human. Is such a message not the result of an action? Yes, it should be seen to be an action result. We must, thus, add another pragmatic dimension to the one described above: the message as a result of an action. The production of this message is to be considered as an action. What else would it be? This is obvious if the producer of the message is a human agent. In this case, the situation is a human-to-human communication. Speaking and writing is action (Austin, 1962; Searle, 1969). Even when the producer is an IT artefact, the production of the message must be a communication action. This thesis, vindicated in the language/action tradition of information systems (e.g. Goldkuhl and Lyytinen, 1982; Winograd, 1988; Dietz, 1994), might perhaps be harder to accept, but this will be elaborated below (Section 4.4).

The sign transfer (i.e. a communication process) is an action. The action within the sign transfer should be seen as a pragmatic aspect. This means that the information system’s production of messages to its environment should also be seen as an action. Pragmatics is not only related to actions based on messages from information systems. The very production of messages (signs) also has a pragmatic dimension, since this sign production should be considered as action.
The sign transfer as action is, with reference to concepts of speech act theory (Austin, 1962; Searle, 1969), considered to be a so-called illocutionary act. This is what is done in speaking. Actions based on the communicated sign are considered to be on the perlocutionary level. In speech act theory, this relates to effects arising from communication: what the listener does covertly or overtly. These two aspects: (a) action within the sign transfer, and (b) action based on the transferred sign; are important pragmatic aspects of information systems and their use.

These two aspects should be both acknowledged and distinguished, as they can be by aid of the speech act notions of illocution and perlocution. As can be seen from Figure 4-1, they are related. Let us elaborate this further. The sign as action might give rise to expectations in the interpreter. These expectations are intrinsically tied to the illocution performed. The sign is a base for action and this action can be seen as fulfilment or rejection of the expectations as well as other perlocutionary effects. When a person A asks a person B ‘Would you bring me a glass of water?’, A performs a request directed towards B. This request action (which is a special kind of illocution) gives rise to expectations directed from A to B. A expects B to perform an action of bringing water. B can react in different ways, declining the request or fetching a glass of water. Refusal or delivery is action in response to the request. These different types of actions are considered as perlocutionary effects of the speech act of the request. The delivery of the water is the effect intended by A.

In the proposed conception of information systems pragmatics, the two action aspects described above are included. Stamper (1996) treated these two aspects within his semiotic theory. In Stamper’s semiotic ladder the illocutionary aspect seems mostly to be related to the pragmatic level and the perlocutionary aspect seems mostly to be related to the social level. In the framework presented above it is not possible to make these kinds of characterizations. The two types of action (action within sign transfer, action based on transferred sign) are both actions and as such they of course involve a pragmatic dimension. Arguably, these two kinds of action involve a social dimension also.

The sign transfer action is a social action since its main purpose is to create a shared understanding between human beings (Habermas, 1984). A social action is that human behaviour which attaches subjective meaning and is oriented towards the behaviour of others (Weber, 1978). Different interpersonal relationships are established because of the actual illocution performed. The action based on the transferred sign is also to be seen as a social action. This is the case even if the listener does not explicitly respond to the ‘initial’ speaker overtly. Even if the listener performs an action (based on the transferred sign) in isolation, this is a social action since the user directs the action in relation to other humans. This seems to be in accordance with Weber’s (1978) classical definition of a social action.

Pragmatics is about action. When we talk about intentionally arranged signs related to action, it seems impossible to exclude a social dimension. The actions of the two kinds are social actions. It is hard to distinguish between the social and pragmatic

---

12 Stamper (1996) explicitly relates the pragmatic level to the illocutionary level: ‘It is also possible to model some aspects of intentional sign use in terms of speech act theory by studying so-called illocutionary verbs’.

13 When describing the social level, Stamper (1996) makes an explicit reference to the perlocutionary level: ‘Each illocutionary act will have a social consequence achieved by the listener(s) performing (a) perlocutionary act(s), which change(s) the social world.’.
stages on the semiotic ladder concerning these types of actions. Producing signs requires actions and such actions are in most cases social actions.

4.4 Information Systems: Action Artefacts in Organizations

4.4.1 Human Action

Human action is about making a *difference*. This is one fundamental thesis of the American pragmatists (Thayer, 1968). Humans intervene in their outside world and create differences. Human action often aims at making material changes. Chopping wood means that wood is transformed into firewood, which can later be put onto a fire to create heat. Action is meaningful and intentional. Humans act in order to achieve ends.

The world of humans is not only a material world. It is also a social world consisting of other humans and their expressions. Communicating is also action (Austin, 1962; Searle, 1969). To utter a sentence, directed towards another human being, is one kind of action. There are differences and similarities between communication acts and material acts. It is, however, possible to use a generic action model\textsuperscript{14} to describe both kinds of action (Figure 4-2).

\begin{figure}
\centering
\includegraphics[width=\textwidth]{action_model.png}
\caption{A generic model of social action (Goldkuhl and Ågerfalk, 2002).}
\end{figure}

We can distinguish between the following categories:

\begin{itemize}
\item Action prerequisites (external and internal)
\item Actor (interventionist)
\item Acting (performance of action)
\item Action result (what is done)
\item Receiving (interpreting) action result
\item Actor (recipient)
\end{itemize}

\textsuperscript{14} This generic action model emanates from earlier work by Goldkuhl and Röstlinger (1999).
In the following, two examples are used to describe the generic model. (1) A communication act of a woman asking her husband to bring in some firewood: ‘Would you please bring some firewood in. I think it is cold in here’. (2) A material act of chopping wood performed by the husband.

The different action categories described above and in Figure 4-2 is used to characterize the two kinds of actions (see Table 4-1 for an overview). This does not mean that there are not important and fundamental differences between communicative and material actions. There are important differences, which will be shown. In communication we transform our own knowledge (as the base) into a linguistic utterance (the result). When doing so we are using language as an instrument. Language must be intersubjective to a large degree, otherwise the communication will not be successful, i.e. no shared understanding will arise. In material action an external base (such as wood) is transformed into a material result (such as chopped firewood). The actor is using an external instrument (such as an axe) to produce results. The utilization of such an instrument (tool) extends the capabilities of the actor. The actor will have different relations towards the different external objects. He or she will have a transformative relation towards the base, a utilizing relation towards the instrument and a producing relation towards the result. Base, instrument and result are thus important categories for describing action. The base is transformed to a result by a competent utilization of an instrument (Engeström, 2000).

Table 4-1: A comparison between a communication act and a material act (example).

<table>
<thead>
<tr>
<th>Category</th>
<th>Communication action (request)</th>
<th>Material action (chopping)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interventionist actor</td>
<td>The wife</td>
<td>The husband</td>
</tr>
<tr>
<td>Intention</td>
<td>To urge the husband to fetch firewood</td>
<td>To fulfil the wife’s request</td>
</tr>
<tr>
<td>Acting</td>
<td>Speaking (request for firewood)</td>
<td>Chopping</td>
</tr>
<tr>
<td>Base</td>
<td>Knowledge about needs</td>
<td>Wood</td>
</tr>
<tr>
<td>Instrument</td>
<td>Language</td>
<td>Axe</td>
</tr>
<tr>
<td>Abilities</td>
<td>Communicative competence</td>
<td>Chopping competence</td>
</tr>
<tr>
<td>Situational comprehension &amp; attention</td>
<td>Perceiving room temperature and non-existence of firewood</td>
<td>Continuously monitoring own chopping</td>
</tr>
<tr>
<td>Result of action</td>
<td>Utterance (request)</td>
<td>Firewood (chopped)</td>
</tr>
<tr>
<td>Recipient</td>
<td>The husband</td>
<td>The wife</td>
</tr>
</tbody>
</table>
| Effects                         | The husband understands and accepts the wife’s request and goes to chop wood | (1) The wife receives firewood to put on the fire  
(2) Heat from the fire |

In the example, there is an oral face-to-face communication action; and in such a situation there is no use of an external instrument. When speaking to a person at some distance we usually require some instrument, often a telephone, to bridge the distance. Written communication implies the use of external instruments, such as a pen or typing equipment.
Base and instrument are two fundamental prerequisites for action. In the case of material action, these prerequisites are external objects. In the case of communication action, these prerequisites are internal elements, i.e. parts of the practical consciousness of the actor. There are other internal prerequisites, both situational and trans-situational. Actions are purposive, i.e. there are intentions associated with the actions (Weber, 1978). However, intentions are not always well defined and deliberate before the performance of the action. Intentions can arise from situations (Norman, 1988; Joas, 1993). Ends and means can be figured out when interacting with the environment. This implies that the situational understanding and the attention of the actor are fundamental for acting (Joas, 1993; Giddens, 1984).

Following the classical model of social action by Weber (1978), actions are also governed by values, emotions and traditions (norms). In accordance with the writing of Weber, it is possible to claim the multifunctionality of action; i.e. action is often governed by all these prerequisites (purposes, values, emotions and norms), which are expressed during action. The request of the wife is of course purposive, but at the same time it can be seen as compliance to a social norm concerning division of work within this family, namely ‘the husband is the one who fetches firewood’. This is also related to the conceived identity of the actors: ‘Who am I – what is my role in this situation?’ The action prerequisites mentioned by Weber (1978) are important, however they are not complete. As can be seen in Figure 4-2 some other important prerequisites are added.

The two actions described together form a social interaction. Of course, we interact with other people through language, but we also interact by producing and delivering material objects to each other. The material action of the husband is assigned by the request of the wife. The wife’s request is self-assigned. In the interaction, different action relationships arise between the participants. The request gives rise to expectations directed towards the husband. The delivered firewood may give rise to emotions of gratitude towards the husband.

There is an important difference in the generic action model between action result and action effects. The action result (what is done) is within the range of the actor, i.e. it is within the actor’s control. The effects are what can arise as consequences of the action. The speaker has no control over the effects connected to the receiver. The listener can understand the utterance in different ways; e.g. they can misunderstand in ways that the speaker could not presume. The listener can act in ways outside the scope of the speaker’s intention. In the communication act case, the action result reflects the uttered request. The effects of the action are the understanding of the receiver and the consequential actions based on this understanding. The result of the material act is the chopped firewood. The effects are what the wife does when she receives the firewood. The reception of a message (oral or written) implies always an interpretation of what is communicated. Even the reception of material objects implies interpretation: ‘What kind of material is this and how can I use it?’ Effects of actions may be both social/non-material and material. This applies to both communication actions and mate-

---

15 Habermas (1984) has contributed to speech act theory by emphasizing that different interpersonal relationships are established between speaker and hearer in communication.

16 These differences can be related to concepts of speech act theory. Searle (1969) distinguishes sharply between what is done within the utterance (the illocution) and what is intended to be done by the listener (the perlocution). The illocution is one part of the utterance, i.e. what is done by the speaker (the result). Effects related to the listener can be named perlocutionary effects.
rial actions. The communication action of the wife – the request – gives rise to both commitments made by the husband and material changes of chopped wood. The material action of delivering firewood to the wife gives rise to social effects (the fulfilment of a performed commitment and the gratitude for this) and material effects (heat from the fire).

This generic model of action can be compared to other models. In this section it is briefly compared to a model by Norman (1988). This is done since the actability concept will be discussed in relation to Norman’s model in Section 4.5 below.

Norman (1988) presents a seven-stage model of human action. The main domain for his action model is humans’ interaction with everyday things. The seven stages are (1) forming the goal, (2) forming the intention, (3) specifying an action (action sequences), (4) executing the action, (5) perceiving the state of the world, (6) interpreting the state of the world and (7) evaluating the outcome. These seven stages can be divided into three phases: preparation (stages 1–3), execution (stage 4) and evaluation (stages 5–7). Norman speaks only of execution and evaluation, but it is also important to distinguish between preparation and execution.

There are some terminological differences between the models. Norman differentiates between the goal and the intention. The differences between these seemingly similar concepts are that goals are to be seen as the ends, and intentions are to be seen as the means. The action model (see above) distinguishes between underlying values and preferences (i.e. more fundamental endeavours and purposes), specific intentions of the action (ends), plans (means for achieving the intentions) and deliberations (which involve considerations concerning both ends and means, i.e. reasons for choosing a certain action and not choosing another one).

Norman’s model can be criticized for having a rationalistic flavour: a strict sequence of preparation, execution and evaluation with predefined purposes before entering into acting. In the suggested model the continuous perception of the world is emphasized, i.e. the world is perceived before, during and after the execution of action. An actor starts from a situational understanding of the action context. This situational understanding determines what ends and means will be chosen. (The wife is thinking ‘It is cold in here, but I could make a fire in the fireplace’). During action, the actor is continuously monitoring his or her actions, the results and effects. This constant monitoring makes it sometimes possible to modify an action during its execution. The actor is attentive in action. When chopping wood it is necessary to be attentive otherwise one can cut oneself. This notion of continuous perception is also acknowledged by Norman (1988) who states that ‘There is a continual feedback loop, in which the results of one activity are used to direct further ones, in which goals lead to subgoals, intentions lead to subintentions. There are activities in which goals are forgotten, discarded, or reformulated.’

Above, some fundamentals of communicative and material action have been described. Not all human action is of these types, which can be labelled intervening action. There are other types of action, such as investigative action and reflective action. These kinds of actions do not aim at external changes, but rather at internal changes of the actor in terms of improved knowledge.

---

17 We think it is possible to combine a purposive view of action with a situational view, i.e. that action is both purposive and situationally constrained. Purposes can arise from situations, but can also be brought into situations by deliberate planning.
4.4.2 Language Acts and Other Communication Acts

The argument so far is that language use, in the form of utterances, can (and should) be regarded as action. This is, however, not the only way communication is performed. The concept of ‘speech act’ can be regarded as a special case of the more general concept of ‘semiotic act’ (Graham, 1998). When two people communicate, they use some inter-subjective conventions of signs and signals. Such signals between performers are called semiotic acts and are always carried by some material substratum. If the substratum is verbal or written then the semiotic act is a speech act (Graham, 1998). This implies that also other substrata might be used to form communication, for example gestures and other forms of ‘body language’ are ways to communicate. The key point is that not just speech acts but all semiotic acts share the properties discussed above. This is the reason why the term ‘communication act’ is preferred instead of ‘speech act’ in this dissertation.

4.4.3 Organizational Action

Organizations create products (goods or services) beneficial for their environment; i.e. for their customers or clients. In doing this, different actions must be performed. There can be material acts with the purpose of producing material results. There can be communication acts with the purpose of producing communicative results (utterances and messages). Material acts and communication acts together form patterns of action. They form not only a language game, but also an ‘activity game’, with relations between communication and material treatment (Goldkuhl, 1995). Large parts of this activity game are recurrent actions and are thus institutionalized in the organization and the practical consciousness of its participants (Berger and Luckmann, 1967; Giddens, 1984). Some of these actions are internal, i.e. directed towards the interior of the organization. Other acts are external, i.e. directed towards its environment.

When using the term organizational action, what is to be meant by this? Does it mean that the organization acts? Or does it mean that the human members of the organization act? Actually, both meanings can be used. Organizations are conceived to have action ability. We can say that organizations act. Is this not a reified position? A view that organizations have been given an ontological status of their own outside the realm of human originators? Organizations are actors, but they are not actors on their own. They always have a human origin and purpose. Organizations are created by humans and for the purposes of those humans. Organizations act always through their human co-workers or through artefacts arranged by humans. They cannot act by themselves, only through humans.

The co-workers act on behalf of the organization. They act in the name of the organization. An act performed by a human co-worker is always dual. It is an action performed by a human being, but is also at the same time an action performed by the organization. Humans act in organizational roles. Their action is representative. They represent the organization when acting. Given this definition, it is possible to say at the same time that an organization acts and that its human co-workers act (Goldkuhl and Braf, 2002).

---

18 Berger & Luckmann (1967) describe the meaning of reification and the problems and dangers of reifying social phenomena, i.e. disregarding the human origin of socially constructed products.
4.4.4 The Nature of Artefacts

When describing human action above, it was pointed out that instruments are often important means for performing actions. To chop wood without an axe is not possible. Humans use artefacts to extend their action ability. Some artefacts improve actions and others even enable certain actions. The modern world is a world crowded with artefacts. They play a dominant role in today’s society (Latour, 1991). Many artefacts go even beyond enabling human action. They are created to be independent performers of action. Such artefacts have dynamic properties, which make it possible for them to function on their own. A modern washing machine can work by itself after a human has started it. The washing machine substitutes the washing work otherwise performed by a human being. The axe does not make anything by itself. It is a tool with static properties. Weizenbaum (1976) uses the concepts of prosthetic tools and automatic machines. A prosthetic tool, like an axe, extends the ability of humans. An automatic machine, like a washing machine, has an autonomous ability to function on its own. There are, however, artefacts that do not easily fit into these categories. How about a car? It has abilities to perform work (like moving), but it does not function totally on its own. A human drives a car. A car requires constant manoeuvring. It is a dynamic tool, but a human must operate it.

As shown in Table 4-2, artefacts divide into these three categories. This classification will be used when discussing information systems based on computers and information technology.

<table>
<thead>
<tr>
<th></th>
<th>Static tool</th>
<th>Mechanical tool</th>
<th>Automatic machine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Way of functioning</td>
<td>Static</td>
<td>Dynamic</td>
<td>Dynamic</td>
</tr>
<tr>
<td>Degree of dependence</td>
<td>Active use by human</td>
<td>Active use and surveillance by human</td>
<td>Autonomous</td>
</tr>
<tr>
<td>Type of action</td>
<td>Tool-supported human action</td>
<td>Human-artefact cooperative action</td>
<td>Artefact action</td>
</tr>
<tr>
<td>Role of humans</td>
<td>Wield</td>
<td>Manoeuvre</td>
<td>Initiate</td>
</tr>
<tr>
<td>Example</td>
<td>Axe</td>
<td>Car</td>
<td>Washing machine</td>
</tr>
</tbody>
</table>

4.4.5 IT-Based Information Systems

IT-based information systems can be used for communication action or for internal or external organizational actions. The IS actions must be well integrated into the activity games of the organization. This means that an IS, as an artefact, must be congruent to the actions of humans and to the overall objectives of the organization.

Three kinds of actions related to information systems may be identified: interactive, automatic and consequential actions (see Figure 4-3).

A user can utilize an information system to perform certain actions. The IS is used as an action medium for the user to perform actions. In interaction between the user and the computer a communication action is performed and the results of this action (i.e. a message) is transmitted through the IS. This kind of action – referred to as an interactive action – is performed with support by the IS and through the IS.
An IT-based information system is not only used as an *interactive mechanical tool*, but also as an *automatic machine*. Usually, a system is provided with an ability to perform certain predefined actions of its own. Based on messages in the system, new messages can be derived and communicated to its environment. These kinds of action are called *automatic actions*. It is important to state that the computer performs these actions in accordance with well-defined rules created by humans. Responsibility for the actions rests upon those humans who decide on the rules to be followed. Arguably, it is possible to talk about automatic actions performed by an IT-based information system in this way, without falling into the trap of reification. In this context, reification would mean forgetting the human origin of information systems and IS action, that we conceive information systems to be independent actors creating actions totally on their own. Such a reifying position is avoided by claiming that information systems perform actions that are ultimately derived from rules specified by human actors. An IS does not find out what kind of actions to perform. The systems’ actions derive ultimately from the rules predefined to the system. Computers are not presupposed to have human properties of consciousness and ethical responsibility.

![Figure 4-3: Different actions related to an information system (Goldkuhl and Ågerfalk, 2002).](image)

Automatic actions of an IS will produce messages (with defined illocutionary and propositional meanings) to its environment, i.e. to some IS users. Such messages can be a basis for actions by these users. Such user actions may be termed *consequential actions*. These actions can be seen as perlocutionary effects in relation to the communication actions performed by the IS.

This view of information systems means a rejection of the view of information systems as mere ‘containers of facts’, as passive providers of information for future actions of human users. Information systems are used for communication among people within and outside the organization. An information system is also an active party in such a communication. It is an artefact capable of performing communication actions on behalf of the organization. Information systems are conceived to be *information action systems*. An IS should have its own pragmatic properties (i.e. be able to perform communication actions with illocutionary intent) and not only bring about pragmatic effects (i.e. humans performing actions based on messages from information systems as ‘perlocutionary responses’). This view is also reflected in my conceptualization of ‘message’.

In traditional ‘infological’ information systems theory (Langefors, 1973), the smallest element of information is the elementary message (the e-message). The basic e-message is a triple \((o, p, t)\) consisting of a reference \(o\) to an object in the universe of discourse, a reference \(p\) to a property predicated to that object and a reference \(t\) to a point in time or a time interval during which the e-message is valid. Similarly, a rela-
tional e-message $e = ((o_1, o_2, \ldots, o_n), r, t)$ is used to assign the objects $(o_1, o_2, \ldots, o_n)$ to the relation $r$. Goldkuhl (1995) argued that the e-message concept must be extended with an illocutionary component to reflect the illocution of the message. By elaborating this further, we arrive at the concept of action elementary message (ae-message). When talking about messages as results of communication actions, ae-messages is what is referred to.

Since an illocution can be (and usually is) associated with a larger propositional content (references and predicates) than a single e-message, an ae-message can be considered as an aggregate of e-messages with an associated action mode representing the illocutionary intention of the ae-message. The key point is that while an e-message is the smallest unit that carries information (or rather propositional content), an ae-message is the smallest unit that carries a communicative action mode. When discussing the pragmatics of information usage in organization, it is not meaningful to dissect information down to the level imposed by the e-message concept. However, when it comes to the design of the persistent storage of information (e.g. a database) it is important to understand the relation between ae-messages and e-messages so that ‘traditional’ approaches can be utilized. It is important, though, to not fall into the trap of disrespecting the coupling between the propositional content and the action mode. In Chapter 5, the relationship between e-messages and ae-messages as basic constituents of information is expanded.

Let us now consider an example to illustrate the concept of ae-message. Suppose that a virtual bookshop is, via their website, offering potential customers a special ‘book of the month’ to buy for a very special price. This on-line offer is an ae-message sent from the bookshop to potential customers. The action mode of the message is a promise that customers can buy the book at the specified price during the specified month. The propositional content might be data on the book’s title, number of pages, author, publisher, price (ordinary and reduced), et cetera. Note that it would require several e-messages to represent this single ae-message’s propositional content and that it would not have made sense to try to connect these separately to its action mode. Doing so would have required another relational e-message referencing the first ones. This would have been to treat illocutions as describable pieces of reality (belonging to the object system) rather than as constituents of communication actions (as parts of the discourse). Langefors (1995) argues, in contrast to what has been said, that ‘It is a design decision whether to represent this [the action mode] in the message structure or in the environment’. Arguably, this is not a matter of design, but rather of fundamentally different ontological standpoints.

4.5 Introducing Information Systems Actability

Information systems actability is a conceptualization of the understanding of information systems as tools for business action and communication. This is in contrast to the misconception that information systems are used only as passive providers of information to be used for future action, i.e. only action based on transferred signs. Actability can be understood as an attempt at a synthesis of the usability perspective as applied within the human-computer interaction tradition (e.g. Shackel, 1984; Preece et al., 1994; ISO 9241-11, 1998; Maguire, 2001) and the language/action perspective (e.g. Winograd, 1988), from an information systems point-of-view.

A central concept for actability is the concept of performance of social communication actions, referred to as elementary actions (e-actions). E-actions result in ae-
messages. As discussed above (Section 4.4.4), there are three kinds of such action related to information systems: interactive, automatic and consequential.

Let us again consider the bookshop example introduced above to illustrate. A user accessing the website and reading about the ‘book of the month’ would be performing an interactive action by purchasing the book, utilizing an order function provided by the Web-based information system. If the user, on the other hand, chose to print an order form and send it by traditional post, this would be a consequential action. Suppose also that the bookshop’s information system collected all orders at the end of the day and sent a supplier order via Electronic Document Interchange (EDI) to a publisher containing all books ordered that day from that publisher; it would have performed an automatic action.

Based on this discussion, ‘actability’ is defined as: an information system’s ability to perform actions, and to permit, promote and facilitate the performance of actions by users, both through the system and based on information from the system, in some business context. (Goldkuhl and Ågerfalk, 2002)

4.5.1 Information System Use-Situations

When talking about the performance of actions in relation to information systems, and thus usage of the system, it is possible to distinguish three different types of IS use-situations:

1. An automatic use-situation occurs when an information system’s ‘ability to perform actions’ is considered. That is, when an IS performs e-actions without any direct human intervention, but according to instructions by, and on behalf of, some human actor. In these situations the IS is used as an automatic machine.
2. An interactive use-situation occurs when an IS is used to ‘permit, promote and facilitate the performance of actions by users … through the system’. In these situations the IS is used as a mechanical tool.
3. A consequential use-situation occurs when an IS is used to ‘permit, promote and facilitate the performance of actions by users … based on information from the system’. In these situations the IS is used as a static tool.

In all three types of IS use-situations, there are three basic components involved: (1) the information system used by (2) an actor to perform (3) e-actions. These three entities participate together in a ternary relationship that forms the use-situation (as depicted in Figure 1-4b, p. 11). To understand these components and how they interact within the ternary relation is the key to understanding actability.

To achieve such understanding it is necessary to comprehend the first component, information systems, as systems for communication actions – that is, as information action systems, which leads to the following definition: An information system is a technically implemented social system consisting of an action potential (a repertoire of actions and a vocabulary), a memory of earlier actions and action prerequisites, and actions performed interactively by the user and the system and/or automatically by the system. (Goldkuhl and Ågerfalk, 2002)

Using ‘technically implemented social system’ is in agreement with the argument of Goldkuhl & Lyytinen (1982) that information systems are not technological systems, but ‘social systems only technically implemented’. To say that information systems have an ‘action potential’ means that any information system per se defines the set of actions that can be performed by using it. This also incorporates the vocabulary used to
express communication actions within it – how to refer to the state of affairs that is talked about. ‘Memory of earlier actions and action prerequisites’ refers to the persistent storage aspect of information systems, which is updated and extended through the performance of e-actions. This is an important feature of information systems used to keep track of what has been said and done through the system and serves as important background for future actions. One can think of this action memory as a repository of messages held by the system. The action memory would typically be implemented as one or several databases, even though it might make use of whatever technological solution is available. In the bookshop example (see above), the action memory would typically be used to store, for example, order messages from customers in an order database. That would be an example of ‘earlier messages’. It might, however, also contain information about what book that will be the book of the month during months to come, which then would constitute action prerequisites. The action memories of an organization’s information systems form an important part of the overall organizational memory. It can be used to inform about who has done what with whom, when, and why. Figure 4-4 depicts this view of an information system as an information action system.

Figure 4-4: Information system as information action system (Ågerfalk, 1999b).

The ‘actions performed …’ in the definition should be understood as the instance level of the system, whereas the ‘action potential’ represents the system’s type level. It is important to include both levels in the definition since neither of them can be understood in isolation from the other. We use information systems to do things and to talk about the world, within the world. Information systems are not separated abstract models representing the state of the world, but parts of the world themselves.
Table 4-3 shows how different parts of an information system belong to these two levels. Note that the instance level consists of two different aspects. Messages represent the state of the system while actions represent a dynamic aspect.

The second component of an IS use-situation, the actor, relates to three different meta-roles: the communicator, the performer and the interpreter. The communicator is the human agent responsible for an e-action in relation to an interpreter (recipient). Sometimes the actual performance of an action is pursued by someone other than the communicator. This is most obvious in the case of automatic use-situations when the information system performs the action and thus plays the role of performer. However, sometimes another human agent performs an e-action interactively on commission of the communicator. This second human agent – acting as a mediator – then plays the role of performer. As indicated above, a human agent, using the system as an information provider for future actions outside the system (i.e. during a consequential use-situation), plays the role of interpreter. Human agents playing the latter two roles, i.e. performers and interpreters, are the ones commonly referred to as users, for example in (ISO 9241-11, 1998). However, this notion of information systems use may be too restrictive. From an actability point-of-view, a communicator, as well as a performer or an interpreter, are users of the information system.

Table 4-3: Description of an IT-based information system with respect to its type and instance levels.

<table>
<thead>
<tr>
<th>Type level</th>
<th>Action potential: rules for what actions to perform, including what types of messages to treat</th>
<th>Rules for action memory (what types of messages to save and the vocabulary used)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance level</td>
<td>Input, intermediate, and output messages (state part)</td>
<td>Performance of interactive and automatic actions (active part)</td>
</tr>
</tbody>
</table>

The third part of an IS use-situation comprises the set of e-actions performed during the use-situation. This set consists of zero or more e-actions, possibly with associated sequence restrictions, that are logically grouped together to form a coherent whole. An interactive use-situation consists of one or several interactive, consequential, automatic and manual actions. It can be defined as: all interactive actions, and intermediate consequential, automatic and manual actions, that are performed adjacent in time by the same performers. A consequential use-situation consists of one or several consequential, automatic and manual actions. It can be defined as: all consequential actions, and intermediate automatic and manual actions, that are performed adjacent in time by the same performers, which are not part of any interactive use-situation. Finally, the automatic use-situations of a system consist of one or several automatic actions and can be defined as: all automatic actions that are performed adjacent in time by the same information system, that are not part of any interactive or consequential use-situation.

It is important to realize that consequential use-situations as well as interactive use-situations might incorporate interaction between humans and information systems. In the latter case, the interaction aims at sending an ae-message through the system. In the former, the aim is to interpret one or several ae-messages in order to perform an action outside the system. It is also important to realize that interpreting earlier messages might also be an important part of an interactive use-situation. Hence, the two
types of usage are tightly coupled, but are still distinguishable. Basically, from an information systems perspective, an interactive use-situation is a consequential use-situation with added complexity. The complexity is imposed by the functionality required to formulate and send messages, and to interpret the effects of the action. Since actions performed during consequential use-situations are always established outside the system, this aspect is not considered part of the IS, but might still be supported by it. What, then, is to be meant by ‘formulating messages’? The concept refers to the fact that before a communication action can be performed, the speaker must formulate the sequence of words to utter (i.e. the preparation of the action). This includes formulating the propositional content of the action and attaching it to an understandable and appropriate action mode (i.e. the illocutionary meaning of the action). To interpret the effects of the action means to comprehend how the action changed the state of the social world in which it was performed (i.e. observing the perlocutionary effects, if possible). An important aspect of actability is that when e-actions are performed through an IS, these two cognitive tasks can be supported by the IS. A performer can, for example, interactively use the system to pick items from a list of merchandise in order to formulate an order message. It is worth noting that this makes the action preparation explicit as opposed to implicit during manual action.

4.5.2 Elementary Interactions
The interactive formulation and sending of ae-messages (and, if possible, the evaluation of their effects) is performed through elementary interactions (e-interactions). An e-interaction is an action at a lower level of abstraction than e-actions. An interacting user performs an e-interaction every time something is done with the interface of the system. Each e-interaction follows a recurring schema consisting of three phases: a user action, an IS action and an interpretation act. The user action represents the user’s direct manipulation of objects on the computer screen, for example clicking a button labelled ‘send’. The IS action represents the information system’s response to the user action, for example sending whatever was to be sent. This phase might include several operations performed by the IS and usually ends with the system giving the user some sort of feedback. The interpretation act represents the user’s effort to understand what the system achieved, by utilizing feedback from the system. This recurring schema is referred to as the **elementary interaction loop (EIAL)** (see Figure 4-5). Note that the loop might be recursively expanded since the interpretation act might require the performance of yet another e-interaction. Hence, a new loop may be initiated in order to complete the one at hand.

During the execution of an instance of such a loop the interactive situation is in a transient state and the loop must be completed before a new e-interaction can occur. Actually, there are three different states involved: an **initial state** ($S_0$) before the user action, a **waiting for system to respond state** ($S_1$) between the user action and the IS action(s), and an **IS action accomplished state** ($S_2$) after the IS action(s).

To decide the appropriate granularity of e-interactions it is possible to follow Graham’s (1998) rule of thumb for deciding the granularity of task-objects. Slightly rephrased, his advice is to decompose interactions until further decomposition would introduce terminology that is not part of the professional language of the business. That

---

19 Note that this should not be mistaken for a rationalistic view on actions (cf. Section 4.4.1). Of course, people do not always consciously think through what to say in advance. However, perhaps they should, at least if what they are saying constitutes important business actions.
is, when computer terminology, such as ‘move the mouse’, is needed to do one more decomposition the appropriate granularity has been reached. Note that allowing for recursive expansion of the loop must be permitted but analysis must explore the possibility that two or more e-interactions might have been regarded as one, in which case these should be separated and treated independently. Note also that interactive use-situations are explicitly derived from the action structures forming business processes and how the EIAL thereby connects business requirements with functional requirements and user interface requirements of the planned IS. This feature is further elaborated in Part III when Actability Design is discussed in more detail.

Both the formulation of ae-messages and the execution of e-actions (i.e. sending of ae-messages) are performed through e-interactions. The latter constitute a special case where the interpretation act includes the evaluation of the e-action as a whole. It is important that a system gives as much feedback as possible about the effects of an e-action. However, because the effects might not be directly observable, the system should at least make the user aware that the e-action has been performed. Note that each e-interaction is ‘triggered’ by the interacting user. Before any e-interactions are performed, the IS might perform one or several initial IS actions to ‘set the scene’. The IS might, for example display some initial information that the user can utilize when formulating a first ae-message. Figure 4-6 depicts the general structure of an interactive use-situation.

Figure 4-5: The elementary interaction loop (Ågerfalk et al., 1999).

Figure 4-6: An ae-message is formulated and sent by an interacting user during an interactive use-situation (Ågerfalk et al., 1999).
Based on information provided by some initial IS actions\textsuperscript{20}, the user formulates an ae-message through a series of e-interactions. Finally the ae-message is sent through the IS (an e-action is executed), causing a business effect. In each e-interaction the user is given feedback from the IS. In the final e-interaction this includes information about the performed action and, if possible, its business effect(s). Note that effects may arise at various ‘distances’ from the performed e-action. Usually, there are intended immediate business effects of an action, such as an offer being made and accepted by a customer. Feedback of such intended effects may be provided by the system. Sometimes the effect materializes at another place or a later point in time, in which cases the feedback may not be possible to give instantaneously. Of course, there may be other social effects of an action totally outside of the control of the original communicator. An example would be a potential customer that receives an offer, considers it an offence, and then files a lawsuit. Measures should of course be taken to avoid such unintended consequences. In most cases (probably all) there are infinitely many effects that may arise as a consequence of an action. In this dissertation the focus is mainly on intended direct perlocutionary effects of actions, as a foundation for understanding IS use.

Within the actability conceptualization of IS usage, actions take place at three different levels of abstraction. E-actions, e-interactions and the three phases of e-interactions represent these levels. In addition, the concept of initial IS actions was introduced above. Figure 4-7 depicts a JSP diagram showing the structure of an interactive use-situation to illustrate how these components relate.

From the JSP diagram we see how an interactive use-situation is initialized by a series of zero or more initial IS actions, followed by zero or more e-actions. For each e-action there is a ‘preparation phase’ during which the e-action is formulated (cf. Section 4.4.1). This is done through zero or more e-interactions, each consisting of a user action, an IS action (which might in fact consist of several operations) and an interpretation act. When the formulation is complete the user enters the ‘execution phase’ and can choose to execute the e-action by performing yet another e-interaction. During this ‘final’ e-interaction the interpretation act includes the evaluation of the complete e-action. Hence, it constitutes the ‘evaluation phase’ as well. It is important to note that the user can choose to abandon the current e-action provided the ‘final’ e-interaction has not yet been initiated. After that, it cannot be cancelled.

Let us return to the virtual bookshop example introduced in Section 4.4.4. The bookshop is, via their website, offering a ‘book of the month’ for a special price. By following a link on their home page, a new page is opened and shows the ‘book of the month offer’ in the frame dedicated to show details of books. A customer (i.e. an interacting user) can choose to click on a button labelled ‘add to shopping cart’ and then click on another button to go to the ‘cashier’. Now the user must provide the system with identification information. This is achieved by clicking on either the button labelled ‘log in’ if already registered, or on the button labelled ‘new customer’. Assuming this is an old customer, when logged on, shipping details are shown in an order document, which also contains information about the book that was put in the shopping cart earlier. To go through with the purchase, the user must now click on the button labelled ‘send order’. A new page is then shown, telling the user that the order is being processed and that the book should be delivered within four days if nothing unusual happens.

\textsuperscript{20} Note that these are optional, as there are interactive usage situations where no initial IS actions are performed.
In the example, every single click on the different buttons constitutes an e-interaction. The message being sent from the customer to the bookshop is an order. The action mode of this order contains two different illocutions. The first is a request to the bookshop to deliver the book in question. The second is a promise to the bookshop that the agreed price will be paid when the book is received. The order ae-message is sent by the ‘final’ e-interaction performed by clicking on the ‘send order’ button. For every e-interaction, the system responded by displaying a different page – helping the user to interpret the actions. Note that the formulation of the order message was initiated as soon as the user added the book to the shopping cart, as the system used this information to give an order proposal. When the user first entered the bookshop site, the ‘book of the month’ offer was shown. This is an example of an initial IS action. Note that the user could have withdrawn the purchase free of obligations, until the moment the ‘send order’ button was clicked. After that the user will (probably) be legally bound to pay for the ordered book.

\[\text{Figure 4-7: JSP diagram showing the action structure of interactive use-situations (Goldkuhl and Ågerfalk, 2002).}\]

Ae-messages can be carried by, and visualized as, documents. ‘Document’ is used here as a generic concept referring to ‘traditional’ paper documents as well as screen documents (e.g. forms, dialogues and windows) and, for example, EDI documents. An

\[\text{\footnote{In reality this means of payment is probably rare. Most websites require credit details up front, so that in fact the bank promises to pay on behalf of the customer at the time of purchase, not the time of delivery.}}\]
interactive screen document is used to interactively manipulate the IS and serves as an action medium in the communication through the IS. In the bookshop example above, every Web page except for the last is an interactive screen document. The last page was a static screen document used only to mediate the order acknowledgment to the customer²².

The actability way of understanding IS usage is influenced by, and should therefore be related to, the ‘seven stages of action’ model of Norman (1988) that was introduced in Section 4.4.1. Since e-actions and e-interactions are both actions, but at different levels of abstraction, they can both be related to the model, as shown in Table 4-4. As Table 4-4 indicates, Norman’s model has been extended with an additional stage 4.5 to show the IS action performed within an interaction loop. It is also during this stage that the possible consequences of the action arise.

Table 4-4: Norman’s (1988) ‘seven stages of action’ model compared with execution of e-actions and formulation of ae-messages by e-interactions.²³

<table>
<thead>
<tr>
<th>Stage</th>
<th>Norman</th>
<th>e-actions</th>
<th>e-interactions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 P r</td>
<td>Forming the goal</td>
<td>Deciding desired perlocutionary effect(s)</td>
<td>Forming the goal</td>
</tr>
<tr>
<td>2 l e</td>
<td>Forming the intention</td>
<td>Choosing action mode</td>
<td>Choosing User action</td>
</tr>
<tr>
<td>3 s p</td>
<td>Specifying the action</td>
<td>Formulating the ae-message</td>
<td>Specifying the User action.</td>
</tr>
<tr>
<td>4 E x</td>
<td>Executing the action</td>
<td>Executing the e-action</td>
<td>Performing the User action</td>
</tr>
<tr>
<td>4.5 e.</td>
<td>Consequences arise</td>
<td>IS action performed</td>
<td></td>
</tr>
<tr>
<td>5 E v</td>
<td>Perceiving the system state</td>
<td>Perceiving the business state</td>
<td>Perceiving the system’s response</td>
</tr>
<tr>
<td>6 a l</td>
<td>Interpreting the system state</td>
<td>Interpreting the business state</td>
<td>Performing the interpretation act</td>
</tr>
<tr>
<td>7 u a</td>
<td>Evaluating the outcome</td>
<td>Evaluating the performed action and the IS</td>
<td>Evaluating what the system has achieved</td>
</tr>
</tbody>
</table>

One of Norman’s contributions in his model, as pointed out by Shneiderman (1998, p. 57), is that he is ‘placing his stages in the context of cycles of action and evaluation’, that is in contrast to other models ‘which deal mainly with the knowledge that must be in the user’s mind’ (Shneiderman, 1998, p. 57). Stages 1 to 3 in Norman’s model correspond mainly to the ‘precondition-cloud’ of the EIAL (Figure 4-5), stage 4 corresponds to the user action of the EIAL, and stages 5 to 7 correspond to the EIAL interpretation act. If we agree to these correspondences, we can see how the EIAL takes Norman’s model even further by taking the system action into account as well (the stage 4.5). This is an important property of human-computer interaction, from a language/action perspective – there is no point in analysing user actions and IS actions separately, without explicitly relating them to each other. Doing so would be to analyse single utterances outside their discourse, which would certainly be meaningless. Consider, for example, the trivial case of an IS action constituted by getting a file name from a disk and asking whether or not the user wants to save changes before exiting. Determining whether such IS behaviour seems rational requires that we know whether

²² That is, assuming that no action potential was made available by the page. It might very well have been, for example, a button labelled ‘print page’. In that case, it would have been an interactive screen document.

²³ Adopted and modified from Ågerfalk (1999).
the user is really trying to exit something or not, i.e. we must know something about the action context.

Norman’s model is general in the sense that it accounts for actions at ‘all’ levels, that is, for single actions as well as for action aggregates (activities). We can therefore use the model to understand both the sending and formulation of ae-messages, by use of information systems. From a strict human-IS interaction point of view, the only thing that differs between the two is the user’s goal and thereby intention. In the first case (e-actions) the goal is to achieve the communication effect and in the latter to create a part of the ae-message (a sub-goal). From a system analysis perspective, on the other hand, it is important to distinguish between them. The importance lies in the fact that a system, in order to be actable, should be clear and understandable regarding which e-interaction that is used to really execute the e-action. Up until that point in the interaction the user can, without obligations, cancel the ae-message; after that point, it cannot be undone.

As discussed above, Searle (1969) distinguishes between four different sub-acts of a speech act: the utterance act, the propositional act, the illocutionary act, and the perlocutionary act. Following Norman’s model, before performing an utterance act, the speaker forms the goal and intention of the utterance as well as formulates the sequence of words to be uttered; ‘specifying the action’ in Norman’s terminology. This is to say that based on the desired perlocution (effect) the speaker chooses illocution (what to do) and formulates the utterance (word sequence) concerning something (propositional content) in his mind before the actual uttering, i.e. thinking before talking. As noted above (Footnote 19, p. 68), this should not be mistaken for a rationalistic view of human action and communication. Certainly, one can perform an utterance without thinking through what to say in advance. The point is that if an important speech act is to be performed, such as a vital business action, a speaker should be aware of what he or she is actually doing by that act. Furthermore, as will be shown below, an IS can (and should, in order to be actable) help the speaker to reach such awareness by supporting both the formulation (during the preparation phase) and the uttering (during execution) of communication acts. Hence, the view is normative, in the context of business action, and not descriptive in a general context of language use.

It was also argued above that a communication act might serve several communication functions. As an implication, Searle’s (1969) concept of the speech act (see page 54) needs to be extended to include several illocutionary aspects and corresponding perlocutionary aspects to be performed within the same communication act. This extension, together with the above discussion of formulation, leads to a concept of ‘communication act’ as consisting of five different parts:

a) A formulation phase.
b) An execution.
c) A propositional content.
d) One or several communication functions.
e) One or several communication effects.

---

24 In Chapter 7 we will elaborate more carefully on the notion of achieving effects. As we shall see, there are other goals than mere instrumental ones (as implied here) to take into account. See also the discussion on multi-functionality in Section 4.4.1, to which this relates.
In the above discussion, the focus has been on interactive use-situations. If a consequential use-situation includes interactive usage of an IS, many of the same characteristics apply, i.e. interactions with the IS follow the structure of the elementary interaction loop. What differs is that the formulation is done implicitly with the support of the system instead of explicitly in an interactive screen document. Furthermore, the execution and evaluation phases are performed without using the system, i.e. manually. It is important to understand that not every consequential use-situation includes interactive usage of an IS. Instead, paper documents are often used as carriers of messages in these situations.

4.5.3 Interactive Use-Situations, Tasks and Use Cases

IS usage is often thought of as performance of ‘tasks’ and interface design thus ought to be based on task analysis (Preece et al., 1994; Shneiderman, 1998). A task is something that a user wants to accomplish using the computer (including software). ‘That task includes the universe of real-world objects with which users work to accomplish their intentions and the actions that they apply to those objects’ (Shneiderman, 1998, p. 61). Sometimes the phrase ‘use case’ is favoured instead of ‘task’ (Lauesen and Younessi, 1998). A use case is defined by Jacobson et al. (1992) as ‘a specific way of using the system … thus a special sequence of related transactions performed by an actor and the system in a dialogue’. The key to task analysis, and use case analysis, is to decompose high-level tasks into smaller ones in order to find the atomic actions constituting the tasks. One problem of task analysis is to choose the most appropriate set of ‘atomic actions’ (Shneiderman, 1998, p. 70). As described above, Graham’s (1998) rule of thumb can be used to decide the appropriate granularity of e-interactions, that is, decompose interactions until further decomposition would introduce terminology that is not part of the professional language of the business.

A further problem of task (and use case) analysis is to find the high-level tasks in the first place (Vemulapalli, 1995). From an actability-perspective, high-level tasks ought to be found and delineated based on the business communication performed through and by the IS. That is why an interactive use-situation was designated above to be a primitive sequence of business actions that consists of all interactive actions, and intermediate manual and automatic actions, that are performed adjacent in time by the same performers (a definition inspired by Holm & Ljungberg’s (1996) notion of ‘conversation’). This way, tasks, and hence human-IS interactions become integrated in the business model and the business context of task analysis is emphasized, thus, potentially, eliminating human-IS interaction design from becoming a separate activity, disconnected from the business development.

4.5.4 Action Memory and Databases

What (hopefully) should be clear by now is that information systems can be viewed as parts of the communication, and thus action that take place in businesses. Suppose that an actor A does something interactively, i.e. communicates an ae-message to an actor B through an IS (A and B might be the same actor, but do not have to be). In this communication the IS acts as mediator of the communication.

As discussed above, an IS should hold a memory of actions already performed (an action memory) as well as preconditions for future actions. This computerized action memory is part of the organizational memory of the business. Since there will always exist manual action as well as tacit knowledge in organizations, it is important to dis-
tistinguish the two. The action memory might be implemented as a traditional database but may as well be constituted by distributed objects or some other technological solution, which might not even be invented yet.

Thus, messages might be stored in the action memory for later retrieval and use in some other context (or the same). Such a message implies an update of the action memory. A message does not, however, necessarily imply an action memory update. It is possible to imagine messages that are merely sent through the IS without making use of previously sent messages, for example e-mail. Some messages might also refer to previously sent messages without actually changing anything in the action memory. In the latter case, the message is related to previously sent messages by its propositional content. See Figure 4-8 for an illustration of this discussion.

Figure 4-8: Information system as communication mediator.

Thus, messages communicated through an IS can be classified in two main categories: (1) messages that are independent of previously sent messages (and thereby do not depend on the action memory), and (2) messages that depend on previously sent messages (and thereby depend on the action memory). Note that if an ae-message that at a first glance seems to be action memory independent constitutes important business action, it should usually be stored in the action memory.

The category of dependent messages (category 2) can be further sub-categorized into messages that (a) merely use the action memory, and (b) affect the action memory. Since ae-messages are products of e-actions, it is possible to classify e-actions as:

1. Action memory independent,
2. Action memory using, or
3. Action memory affecting.

This distinction might be useful when constructing an information system. It will, for example, give a hint about what kinds of components to use when constructing the user interface: static, data aware, et cetera. This distinction is also used in Chapter 14 when the Internet-based information system is characterized from the perspective of actability.

4.5.5 Actability Features of Information Systems

All documents have an action potential, i.e. they can be used for action. In the case of interactive screen documents the, action potential is materialized as the functionality offered to the interacting actor at any given point in time (i.e. the possible e-interactions to initiate) by, for example, screen items to click.

An interactive screen document’s action potential varies over time as the result of (1) the (type of) interactive use-situation at hand and (2) the state of that situation. A
‘Product information’ interactive document might, for example, offer different action potential during the interactive situation ‘Create customer order’ and the situation ‘Stocktaking’. During ‘Create customer order’ the same document might also require that some particular customer details have been entered (a certain state is reached) before the order can be registered. Such changes in action potential can, and should, be visible to the user, which could be done by, for example, disable buttons that cannot be used in a certain state.

From the above it is possible to conclude that in order for an IS to be regarded as actable during an interactive use-situation it should be able to help (permit, promote and facilitate) an interacting performer to:

1. choose which communication action to perform (at action level);
2. formulate the propositional content of the action and attach it to an understandable and appropriate action mode (at interaction level); and
3. execute the communication action; and finally to
4. interpret and evaluate the action and its immediate business effect(s).

The visibility of action potential is similar to Norman’s (1988) concept of affordance\(^{25}\), i.e. ‘the perceived and actual properties of the thing, primarily those fundamental properties that determine just how the thing could possibly be used’. The point of taking affordance and hence action potential visibility into account is to promote designs from which ‘the user knows what to do just by looking: no picture, label, or instruction is required’ (Norman, 1988).

Furthermore, the IS should be able to help (permit, promote and facilitate) an interpreter to acquire sufficient background to perform the intended action during a consequential use-situation.

Finally, the IS should be able to make the communicator feel confident that the performer (human or automaton) really performs the intended action and that the resulting message(s) reaches the intended interpreter(s).

Any use-situation must be understood within the context of doing business with some other party. In a business setting, the interpreter (recipient) of an ae-message must trust what is said (and thus communicatively done) by a speaker (communicator). To understand the meaning of ‘trust what is said’ we can turn to the ‘universal validity claims’ proposed by Habermas (1984). He argues that such claims are raised by any communicator and presupposed by the communicator to be accepted by the interpreter. Successful communication implies that the listener must both comprehend the action and accept it as valid. That is, the communicator and the interpreter must agree on the communication act in order for it to be successful. To reach such mutual understanding a communicator raises four universal validity claims (ibid.): (1) a claim of comprehensibility, (2) a claim of truth, (3) a claim of rightness, and (4) a claim of sincerity.

The first claim, (1) the claim of comprehensibility, means that a communicator must be able to formulate a grammatically correct, comprehensible sentence. In the context of the bookshop example above this means, for example, that the ‘book of the month’ on-line offer must be expressed in a way understandable by potential customers. This can be achieved in several different ways, for example by conforming to accepted standards, protocols and user interface design guidelines. This relates to the

\(^{25}\) The concept of affordance was originally constructed by Gibson (1977; 1979) and elaborated in the context of user interface design by Norman. See Bærentsen and Trettvik (2002) for a critical discussion about the concept and how it has been used in Human-Computer Interaction.
‘permit, promote and facilitate’ in the definition of actability (see above). The second claim, (2) the claim of truth, means that an action must refer to the true state of affairs. In the on-line offer example, this would, for example, imply that there actually are books to be delivered as expressed in the offer and that these books are correctly described. The third claim, (3) the claim of rightness, means that a communicator must be able to establish an interpersonal relationship in the right way according to accepted social norms. For example, the bookshop must be able to act according to applicable legal and cultural norms of trade. The fourth claim, (4) the claim of sincerity, means that a communicator must have sincere intentions, for example, that the bookshop actually intends to sell its books to a visitor.

From this it is possible to conclude that if an IS is to be regarded as actable its users must be able to both comprehend it syntactically and semantically and accept its underlying intentions as true, sincere and normatively right. Following Habermas (1984), when communication acts are performed based on mutual understanding such that these validity claims are considered, we say that the communication act is a communicative act.26

4.6 Implications for Information Systems Work

The concept of actability has important implications for the way information systems are designed. It implies a shift from the traditional focus on data structures (i.e. syntactic and semantic aspects) to take the systems, actors and their actions as the point of departure for design. It is by understanding the structure of an organization’s actions and hence its communication patterns that it becomes possible to design actable information systems. This also means a shift from focusing on the functions of organizations and information systems. The action structures of an organization form its business processes. Thus, it is not adequate to talk about business functions that require an information system to have various functionalities for it to be effective and efficient. Instead, information systems design should focus on what the actors do (communication actions and material actions) to create value for the organization’s clients. Designing an IS then implies designing its action potential so that it matches the requirements of its users. Designing for actability is a holistic approach where the value-creating activities of an organization as a whole are considered – there is no point in designing information systems isolated from their action context.

4.7 Summarizing the Presented View

There is an urgent need to create a deeper understanding of information systems in organizational settings. Information systems are technically implemented, but they cannot be understood as merely technical objects. Their special character is that they are formalized sign systems and as such are used for human communication. A semiotic perspective, emphasizing the sign and communication character of information systems, is necessary when establishing an organizational theory of information systems. This chapter has been an investigation into different aspects of information systems from a semiotic standpoint with emphasis on the top stages of the semiotic ladder (Stamper, 1996) – the social and pragmatic aspects. A key concept for such an understanding of social and pragmatic aspects of information systems has been presented:

26 ‘“Acts of communication” should not be confused with what I have introduced as communicative action’ (Habermas, 1984, p. 295)
Information systems actability. Also some subjects relating to meaning, structure and media have been discussed.

Two different types of actions related to signs have been identified:

- Action within the sign transfer
- Action based on the transferred sign

These two types of actions can be related to the conceptualization of information systems actability and the three types of IS use-situations. Action within sign transfer is what is performed during interactive and automatic use-situations. Action based on the transferred sign is what is performed during consequential use-situations and sometimes during interactive use-situations.

An analysis of the role of messages (sign constellations) within information systems has been conducted. In the analysis speech act theory was used as the main source of inspiration. This led to the concept of the ae-message consisting of an action mode (type of illocution) and a propositional content. Ae-messages are results of communication actions referred to as e-actions. Input and output messages of information systems are realized through documents. Interactive screen documents play a special role: not only are they a result of action, they are also used in interactive use-situations for several purposes. Such a document can aid the user to formulate the ae-message (the propositional content) and attach it to an appropriate action mode. In addition, it is also an action medium for the execution of the e-action.

Information systems actability seems to be an important conceptualization for understanding the use of information systems in businesses. It makes it possible – within one theoretical framework – to relate different aspects (such as technical, cognitive, behavioural, linguistic and organizational) to each other. This is possible because of the basic pragmatic perspective. Starting from human action, a coherent description of social action, organizational action and artefacts usage as a basis for formulating the concept of information systems actability has been given.

The Oxford English Dictionary defines the term ‘actability’ as ‘capability of being acted’. It is the substantive form of ‘actable’, which is defined as: ‘capable of being acted or carried out in practice’. The term is traditionally used in the context of theatre and drama, and Leven (1995) was probably among the first to use it when discussing information systems. His definition refers to a slightly different concept though. According to Leven’s definition, actability is what you get when raising your eyes above the computer screen and take the value-creating, and customer-centred, activities of a commercial firm as starting point for IS evaluation. This is in accordance with the view of actability, but a too restrictive notion to use as a definition. You do not need any customers to be able to talk about actability. The point is not whether the information system creates value for the customer; but that it creates what the actors intend to create. In a commercial firm, an actable IS would probably, directly or indirectly create value for the customer. But in, say, a university, with a very unclear notion of ‘customer’, a system to keep track of whose turn it is to make coffee would still be actable if it actually eased the professors’ burden of coffee making. Leven distinguishes actability from usability by stating that usability is about the relationship between the user and the system whilst actability is about the relationship between the client (customer) and the user-system constellation. As summarized in the A³ model (see Figure 1-4, p. 11), actability is, in this author’s view, about the ternary relationship between actors (i.e. users, which might be customers), information systems and business actions (which might involve customers).
Above actability was defined as ‘an information system’s ability to perform actions and to permit, promote and facilitate the performance of actions by users, both through the system and based on information from the system, in some business context’.

The statement that information systems have the ‘ability to perform actions’ implies that they are seen as agents performing actions on someone else’s behalf. In this context, actions are thought of as speech acts (communication actions), i.e. performances of social actions by use of some inter-subjective conventions of signs and signals. Someone performs an action to effect a change in the business, which creates an action relationship between the speaker and one or several listeners. It is important to note that IS actions are always ultimately derived from predefined rules and that a human actor is always ultimately responsible for them.

The statement that information systems should ‘permit, promote and facilitate the performance of actions by users’ implies that an IS should not only be easy to use and explicit about actions that it is possible to perform but should also encourage users to benefit from acting through it.

User actions could be performed ‘through the system’, i.e. a user performs actions with an IS as a vehicle for communication. However, ‘based on information from the system’ implies that an IS is used to create action possibilities. It is important to note that the term user refers both to interacting users and to actors who use the system without physical contact with it – for example, actors who have the system, or an interacting actor, perform actions on their behalf. Hence, a manager of a sales department who is responsible for the business actions of his employees is a user of a sales support system with which his employees perform actions interactively, even though he might never see the system himself. Even someone using a paper document created by the system as a basis for manual action is regarded as a user.

The actability possessed by a certain IS is always related to a particular ‘business context’ in which the IS is used. The business context includes actors’ pre-existing knowledge and skills regarding both the IS and the business tasks performed. Therefore, actability is not a static property of an IS but depends on the social structures surrounding it.

4.8 Criticisms of Speech Act Theory

Let us now pay attention to some of the criticisms that has been put forward against using speech act theory as a foundation for the design of information technology before leaving this chapter to dig deeper into important areas of actability. Even though the Language/Action Perspective has gained much attention, and proved successful over the years, it has also been criticized. Ljungberg and Holm (1996) have investigated the pros and cons of adopting speech act theory as a foundation for design and identified two main sets of criticisms. The first set of criticisms concerns the problem of theoretical abstraction and can be broken down into two ‘lines’ of criticism: (a) the insufficiency of any theoretical abstraction, and (b) the insufficiency of speech act theory in particular. The second set of criticisms concerns the problem of a rationalistic design of work.

As it seems, the second set of criticisms and the second line of the first set in this classification are intertwined. In all modelling efforts some abstractions are made. As pointed out earlier, any method or model directs attention to certain kinds of phenomena (see Section 2.3). This was claimed as an argument for the use of systems devel-
development methods in that they help analysts to identify important aspects of the modelled piece of reality. Of course, if the underlying theory is not able to capture the important aspects, then we are in deep water. One can also use the arguments of Wittgenstein and conclude, from a philosophical point of view, that any attempt to produce a theory of meaning fails, simply because it is a theory. However, as pointed out by Ljungberg and Holm (1996), there is a difference between a descriptive theory used by passive observers (such as philosophers) and a normative theory used by active designers (such as systems developers). Ljungberg and Holm (1996) further argue that ‘The crucial question concerns what needs to be articulated about work and communication to improve current praxis in the development and usage of IT’. These kinds of criticisms are not specific to speech act theory and the Language/Action Perspective. Rather they can be directed towards any theory, method or model for systems engineering. In order to handle the complexity of information systems development, we need to abstract and to use models. The critical issue thus seems to be to use theories that direct attention to the ‘right’ phenomena and, most important, that developers and other system stakeholders understand the importance of, and discuss and reflect upon, underlying theoretical foundations, and that an inter-subjective understanding of ‘right’ is reached.

One criticism belonging to the ‘second line of criticisms’ of the first set (i.e. the insufficiency of speech act theory) according to Ljungberg and Holm (1996) is the question of ‘discourse versus conversation’. This topic is further elaborated by Holm and Ljungberg (1996) who discuss whether systems development based on speech acts imposes discourses on businesses when conversations are what to strive for in order to gain flexibility. To make this distinction they resort to two different linguistic schools for analysis of texts: discourse analysis and conversation analysis. From this point of view, a discourse is basically a larger part of a text than the sentence. A discourse is thus constituted by a pattern of discourse-parts (i.e. speech acts) into a well-defined discourse schema. A business model based on speech acts is thus claimed to be restricted to the particular action sequences imposed by the globally managed discourse pattern (or schema). The actors participating in a conversation, on the other hand, locally manage the conversation. Each actor can choose to continue (answer) a course of utterances, or to move the conversation in another direction. Alternatively a speaker can choose to invite another actor to continue the conversation, or start a new one. The problem, according to Ljungberg & Holm (1996), is that work in conversation analysis has shown that conversations are not usually structured in the way indicated by the conversation for action schema. Rather, conversation is usually built up of pairs of utterances as, for example, question-answer, offer-acceptance, et cetera.

Holm and Ljungberg (1996) further argue that business rules often require that a globally managed discourse is necessary but that the use of information systems within such discourses should be regarded as conversations. They propose a design method (the Commodious method) that treats business processes as discourses and human-human-computer interaction as conversations that conform to conversation schemas.

The distinction between discourse and conversation is probably valid in the context of human-human interaction. Since we are dealing with machines with finite instruction repertoires there can be no ‘true’ conversations (or Wittgensteinian language games for that matter) when it comes to human-computer interaction. At every given point in time there is a well-defined finite set of user actions possible to perform. That

27 Note that with the distinction between discourse and conversation, the conversation for action schema of Action Workflow describes a discourse and not a conversation.
is, in every interactive use-situation there is a pattern that the course of actions has to conform to. Thereby all interaction between humans and computers conforms to a discourse and the discourse is defined during systems development. Of course, there are often good reasons for keeping action sequence restrictions at a minimum and thereby providing more flexible user interfaces. At other times, business rules require that certain action sequences are either fulfilled or abandoned. In other words, interaction between humans and computers always conforms to a discourse but a rule of thumb would be to keep users as unaware of that as possible, and strive for a conversational discourse as illustrated in Figure 4-9.

Another criticism brought up by Ljungberg & Holm (1996) is that of the multifunctional nature of communication acts. This issue, which has been handled within the concept of IS actability, was discussed above and is further elaborated in Chapter 5.

A further criticism concerns the ignorance of the information content of speech acts in the conversation for action schema. The focus is on who is communicating when, and not on what is being communicated. This is probably an (over) reaction against the “traditional” descriptive approaches to information systems. As is discussed in Chapter 6 and in Part III of the dissertation, the propositional content plays an important role within actability, as well as in, for example, the works of Searle (1969; 1979).

![Figure 4-9: Flexible IT support as a conversational discourse.](image)

Obviously there are issues in applying theories of communication action to the field of information systems that it is important to be aware of and to actively discuss. The important thing is never to believe that any theory is all-inclusive and the theory with a capital T. Just as businesses change so does our understanding of doing business and designing information systems. The concept of actability shall therefore be viewed as representing what is known at the time of writing of this dissertation, and will (hopefully) continue to evolve.
Chapter 5

Messages in Information Systems


5.1 Introduction

What should be clear from Chapter 4 is that information and communication are complex phenomena, especially if the social action context is taken sufficiently into account. Nonetheless, deciding on the elementary units by which to understand and analyse these phenomena is extremely important when communication is to be performed through and by means of a computerized information system (IS). This is so because important properties of the social context that make communication successful, which are readily available in a face-to-face situation, must be formalized and encoded in the system. In a face-to-face situation, participants (speakers and listeners) share the same physical environment, see and hear each other, and perceive each other’s actions instantaneously (Clark, 1996). On the other hand, when introducing an IS these properties do typically not hold and the design of the IS must compensate for the resulting lack of expressiveness.

The traditional and still predominant view of the concept of information within the field of information systems has been characterized as a ‘contents view’ (Goldkuhl, 1995), representing a ‘descriptive perspective’ (Holm, 1996). In such a view, the common interpretation of ‘information’ seems to be ‘linguistic sentences with the purpose of informing people’ (Goldkuhl, 1995, p. 64). In this view, information systems are regarded as repositories, storing representations of objective facts about the real world. One influential Scandinavian theory of information systems embodying such a descriptive perspective is that formulated by Langefors (1973; 1995). His ‘infological’ theory has had a great impact on information systems research in Scandinavia and throughout the world (Kuutti, 1995; Iivari and Lytyinen, 1998). The infological theory bears much resemblance to other important theories within the information systems field, such as the relational model of data (Codd, 1970) and the entity/relationship approach (Chen, 1976), as well as more recent object-oriented approaches (Jacobson et al., 1992; Booch et al., 1999).

Several information systems scholars working within the so-called language/action perspective have challenged the descriptive approach to information and information systems (e.g. Goldkuhl and Lytyinen, 1982; Winograd and Flores, 1987; Dietz, 1994; 2001). One such effort is that of Goldkuhl (1995), who analyses one of the central components of Langefors’ theory, the elementary message (e-message). In Langefors’
theory, the e-message is used to represent an elementary fact about the world, such as the price of a certain product. In his analysis, Goldkuhl (1995) concludes that the concept of the e-message, as the most basic information-carrying unit, should be extended to include not only references to things in the world (such as products and prices), but also the intentional aspects of language use (why and in what context the price is attributed to the product), in order to view information as action and communication. Goldkuhl’s arguments are in line with theories of speech acts (Austin, 1962; Searle, 1969; 1979) and communicative action (Habermas, 1979; 1984), which suggest that language is used not only to describe things, but also to perform actions (see Section 1.3.3, Chapter 4 and Sections 5.2 and 5.3).

Building further on Goldkuhl’s (1995) analysis, the concept of an action-elementary message (ae-message) was introduced in Chapter 4 that reflects this proposed extension. An ae-message is said to consist of one or several e-messages augmented with an action mode (an illocutionary component representing communicational intents). The key point is that, while an e-message is the smallest unit that carries information (propositional content), an ae-message is the smallest unit that carries an action mode. An ae-message is described as the result of a performed elementary communication action (e-action), which can be thought of as a speech act (Searle, 1969; 1979) or communication action (Habermas, 1984) performed by use of a computerized information system (IS).

The aim of this chapter is to present the concept of the ae-message in more detail, and this includes a clarification of how it can be interpreted. It will be shown that the ae-message can be viewed in different ways, of which two are both in some sense equally intuitive. It will also be shown how the ae-message relates specifically to the use of information systems and the notions of input messages and output messages in order to establish a solid foundation for the understanding of information systems as information action systems.

5.2 From Description to Language/Action

As the point of departure for discussion, Goldkuhl’s (1995) example will be followed and the infological theory of Langefors (1973; 1995) will be used as representative of the descriptive perspective on information and information systems. In this theory, a central concept for the understanding of information is that of the elementary message (e-message). For Langefors, the e-message constitutes the smallest structure that carries information. He states that ‘while an elementary message has a certain information content, or semantic content, nothing smaller than an elementary message has’ (Langefors, 1973, p. 231). Langefors has described the concept of the e-message in several slightly different ways. For this discussion, a version in which the e-message structure is said to consist of four basic terms needed to give information about a property of an object will be used (Langefors, 1973, p. 320):

a) the identity of the object,
b) the kind of property (i.e. the attribute) we want to specify for the object,
c) the specification of that property for that object (i.e. the value of the attribute), and
d) the point in time at which the information is valid.

This structure is often described as a triple \( (o, p, t) \), where \( o \) refers to an object, \( p \) to a value of an associated property (i.e. \( b \) together with \( c \), in the list above) and \( t \) to a
point in time\(^{28}\). Evidently, the concept of the e-message\(^{29}\) is closely related to that of the ‘tuple’ used to form the basis of the relational model of data (Codd, 1970), and to that of the ‘regular entity’ in the entity/relationship approach (Chen, 1976), as well as to that of the ‘entity object’ in object-oriented theory (Jacobson et al., 1992). This resemblance is also noted by Langefors (1995) and Goldkuhl (1995).

When Langefors speaks of information, he refers to ‘something we get to know … knowledge of some sort’ (Langefors, 1995, p. 105). In this view, information is, somewhat vaguely, considered similar to knowledge and related to the process of increasing one’s knowledge – that is, of becoming informed. Throughout his work, Langefors (1973; 1995) is explicit on the point that a certain set of data (in the form of e-messages) cannot, in itself, provide information. To clarify this point he introduces what has become known as the infological equation, \(I = i(D, S, t)\), which states that the information (I) that is communicated by a set of data (D) is a function (i) of the data, the ‘receiving structure’ (S) and the time interval (t) during which interpretation takes place (Langefors, 1973). Information is the result of someone with a certain pre-understanding (the receiving structure) interpreting data given a certain amount of time. After interpreting the message, the agent may or may not perform some action based on that interpretation.

In this view, the main topic of interest is the semantics of information: how some human agent interprets and understands a semiotic sign, as represented by an e-message, which is mediated by an IS. An important goal from an IS development point of view is to achieve a correspondence between the representations (the e-messages) in the IS and the corresponding ‘elementary facts’ in the world that they describe. Another goal is to gain inter-subjective understanding between the users of the IS, so that everyone interprets the representations uniformly.

The descriptive perspective on information and information systems has been challenged by several information systems scholars working within the language/action perspective (e.g. Goldkuhl and Lyytinen, 1982; Winograd and Flores, 1987; Dietz, 1994). One significant critique concerns the way pragmatic aspects of information are handled. Large parts of Chapter 4 was concerned with how the common understanding of the pragmatic aspect of information systems seems to take the form of the kind of ‘\textit{ex post}’ action’ inherent in the infological equation. That is, the pragmatic aspect of information systems seems to be restricted to actions performed by humans after receiving messages from information systems: messages as signs for action.

To correct this purported misconception, Goldkuhl (1995) argues that the e-message needs to be extended with an explicit action component in order to take account of the language/action of which it is a part. This position is mainly derived from the theory of speech acts, originally formulated by Austin (1962) and later refined by Searle (1969; 1979). According to speech act theory, a social action in the form of a speech act consists of two basic components: a propositional content and an illocutionary force, depicted by Searle (1969) as \(F(p)\). The propositional content \(p\) represents what is talked about, the semantics of the utterance. The illocutionary force \(F\) (or rather the illocutionary point, which is only one part of the illocutionary force), represents

\(^{28}\) Note that the time \(t\) can also be understood as a time interval during which the information is valid (Langefors, 1995).

\(^{29}\) Note also that, as described in Chapter 4, there is another type of e-message structure used to represent relationships between objects. The distinction between the two is not important for the discussion in this chapter; see Sundgren (1973) for further details.
what uttering the sentence does in relation to potential listeners. A classic example by Austin (1962) is that of the wedding ceremony. When the bridegroom utters 'I do', he does not do so to describe what he is doing or to state that he is doing it. Rather, he is actually performing the act of taking the bride as his lawfully wedded wife. The point is that speaking is not only about description, but also about performing action. People do things by speaking: promising, questioning, directing, and so on.

The concrete solution proposed by Goldkuhl (1995) was to extend the concept of the e-message with explicit references also to the actor communicating the information and the type of action used in the communication, the illocutionary point.30 This idea was refined in Chapter 4 into the concept of an action-elementary message (ae-message). An ae-message is understood to be a collection of e-messages31 augmented with an action mode representing the illocution associated with the collection. From a descriptive perspective, the e-message constitutes the smallest unit that carries a meaning. From a language/action perspective, the smallest unit that carries information is that which is associated with an action mode – that is, F(p). Langefors (1973, p. 231) argues that 'if we take an elementary message minus one of its terms it then has no semantic content'. Accordingly, one can argue that if we take an ae-message minus one of its terms (i.e. minus the set of e-messages or the action mode), it would not have any pragmatic meaning in the social context of its use.

From a language/action perspective, the social context in which a message is used is imperative in understanding the pragmatic meaning of a message (Winograd, 1988; Clark, 1996). As indicated above, the infological equation stresses the importance of the interpreter. However, to take the social context fully into account we must go beyond that notion and see that any utterance is related to both speaker and listener as well as to the context (Vološinov, 1985). A message is communicated by someone with a certain intention and has certain effects on a listener. At the same time, it creates obligations on the part of the speaker (Bühler, 1934; Habermas, 1984). This condition is described vividly by Vološinov, who asserts: ‘Orientation of the word toward the addressee has an extremely high significance. In point of fact, word is a two-sided act. It is determined equally by whose word it is and for whom it is meant. As word, it is precisely the product of the reciprocal relationship between speaker and listener, addressee and addressee… A word is a bridge thrown between myself and another’ (Vološinov, 1985, pp. 52–53).

Unfortunately, however necessary it may be, taking the social context into account complicates the analysis of information and information systems. While it is straightforward to map semantically the propositional content of an utterance onto one or several e-messages, placing this content within a context of communicating actors, and thus relating the utterance formally to a speaker and one or several listeners, is far from straightforward. If these pragmatic aspects are not sufficiently taken into account dur-

30 As pointed out in Chapter 4, Langefors argues that ‘it is a design decision whether to represent this in the message structure or in the environment’ (Langefors, 1995, p. 116). It was concluded that such an approach is inappropriate in that it treats illocutionary forces as describable pieces of reality (belonging to the object system) rather than constituents of communication action (as parts of the discourse).
31 This collection of e-messages corresponds to what Sundgren (1973) refers to as a consolidated message (c-message): ‘a c-message could be any combination of e-messages. A priori any message … will be considered to be a c-message, which, after analysis, may be broken down into e-messages.’ (Sundgren, 1973, p. 97)
ing systems analysis and design, and, consequently, not properly represented in the systems, we run an obvious risk of ending up with systems that, for example, fail to provide relevant information, that users do not understand how to use, and in which users cannot trace responsibilities for information, actions and commitments made – that is, un-actable systems.

5.3 Actions and Messages in Context

As was argued in Chapter 4, the view of messages as signs for action is too restricted a notion for understanding the pragmatics of information systems within a business context. Pragmatics should not be restricted to action based on transferred signs, but should be concerned with action within sign transfer as well: messages are signs of action as well as signs for action. Attention should be directed towards the listener as interpreter of the sign, as well as towards the speaker as producer of the sign (Clark, 1996). This is in line with Vološinov’s (1985) argument (see above) and the important thing is that the listener’s future actions (‘new bridges thrown’), based on the interpreted sign, constitute a pragmatic aspect of the sign, as does the speaker’s creation of the sign in the first place.

5.3.1 The Concept of a Speech Act

To say something is, according to Austin (1962), to perform three simultaneous acts: one locutionary act, one illocutionary act and one perlocutionary act. The locutionary act is the act of uttering certain words in a certain sequence with a certain sense and reference (i.e. meaning). When performing a locutionary act an illocutionary act is also performed – that is, we do something by speaking, for example, making a promise or giving a warning. Doing something will probably also (usually) have some effect on the hearer of the uttered words, for example, changing her knowledge or causing her to act in response. Causing these effects is to perform a perlocutionary act. Searle (1969) adopted Austin’s concepts of illocutionary and perlocutionary acts, but refined the concept of locutionary act. This led to the concept of a speech act as consisting of four different sub-acts, discussed in Chapter 4:

a) uttering words = performing utterance acts,
b) referring and predicating = performing propositional acts,
c) stating, questioning, commanding, promising, et cetera. = performing illocutionary acts, and
d) causing effects in hearers = performing perlocutionary acts.

Searle is explicit about the first three elements (a), (b) and (c) not being separate things that a speaker does simultaneously. Likewise, (a) and (b) are not means to achieve (c). Rather, “utterance acts stand to propositional acts and illocutionary acts in the way in which, e.g., making an “X” on a ballot paper stands to voting” (Searle, 1969). This is different from the perlocutionary act (d), which is not really an act at all (Allwood, 1976; Eriksson, 2000); rather, it is the effect that the speech act has on the hearer. While the speaker in performing the speech act completes the first three, the actual perlocution does not lie within the control of the speaker. In fact, the distinction between illocution and perlocution can be questioned on the grounds that they both seem reducible to concepts of ‘intention’ and ‘result’ (Allwood, 2000). It seems more promising to think of the illocutionary act as a means to achieve eventually the desired
(perlocutionary) effects, and therefore the terms ‘communication function’ and ‘communication effect’ are used in the following to refer to these two aspects of the model.

5.3.2 The Multi-Functionality of Speech Acts

Several authors have criticized Searle’s (1969; 1979) notion of the speech act on the grounds of a purported underlying assumption that one speech act is always potentially associated with one (and only one) illocutionary act (e.g. Allwood, 1976; 2000; Bowers and Churcher, 1988; De Mickelis and Grasso, 1994; Ljungberg and Holm, 1996). Instead, the multi-functionality of speech acts has been stressed. One aspect of multi-functionality is that a single utterance, referencing a specific propositional content, can be directed towards different listeners with different illocutionary (and perlocutionary) intent. Furthermore, a speech act can be directed towards a single listener with multiple intents. As an example, consider the case of a professor proclaiming ‘let us have a five-minute coffee break’. To most students, this will probably be interpreted as a promise: now there will be a break for five minutes without lecturing, during which coffee may be consumed. To the teaching assistant sitting in the corner, it might instead be a directive to prepare the overhead-projector so that the professor can show the slides he prepared last night. This would be an example of the first case of multi-functionality: two illocutionary acts directed to separate listeners associated with one utterance act. Let us assume that the professor instead proclaimed a thirty-minute break. In this case, the teaching assistant might very well consider this as both a directive to prepare the overhead-projector and a promise allowing for some coffee sipping on his part as well. This would be an example of the second case: two different illocutionary acts (within the same speech act) directed towards one single listener.

As suggested above, the result of a speech act can be thought of as a sign – as the bridge thrown between speaker and listener. Note that this result should not be confused with the perlocutionary act (i.e. with the communication effect that the speech act has on listeners). It is an action result in the sense discussed in Section 4.4.1. That is, a semiotic sign created through the speech act, to be interpreted by listeners. In the context of information systems, there is clearly a need for such a concept, since the interpretation may not occur instantly as in face-to-face communication (Clark, 1996). Rather, messages are mediated by, and typically stored in, the IS for later retrieval and interpretation – as signs of action. This sign notion of a speech act is illustrated in Figure 5-1.

According to Weber (1978) in his classical definition of social action, the term action refers to the human behaviour that attaches subjective meaning, and the term social refers to the conditions wherein one takes into consideration the behaviour of others. Consequently, social action is the action (including inaction) that is oriented to the behaviour of others. Based on these concepts, a speech act (or communication act) can be thought of as a social action that changes the deontic state of the world – that is, an action oriented to the behaviour of others that creates obligations. Such an obligation may be directed towards the listener, for example, when the speaker intends to make the listener behave in a particular way. As suggested above, it may also be directed towards the speaker, in which case the speaker makes a commitment to behave in a particular way. In many cases, the obligation is mutual: it is directed towards both speaker and listener. In such cases, there is an additional kind of multi-functionality involved. Consider, for example, a business offer, which is both a request by the supplier directed towards the customer and a promise on the part of the supplier to sell the
offered products should the customer accept the offer. Actually, following Bühler (1934), all communication acts are oriented to both speaker (through the expressive function of speech) and listener (through the appellative function). This aspect will not be distinguished since the important thing is that, following Weber (1978), all social actions are oriented to the behaviour of others and are usually purposive. Therefore, we can assume that all speech acts have at least one, usually intended, function directed towards at least one intended listener.

Neither (a) so-called indirect speech acts (Searle, 1969) – that is, when we do not say what we wish to say in the most direct manner, nor (b) speech acts performed with a ‘hidden agenda’ – that is, so-called ‘strategic action’ (Habermas, 1984; Ljungberg and Holm, 1996) will be distinguished. The reason for this is that from an IS design perspective, it is (a) imperative to design systems in which the meanings of possible actions are clear and visible (this will be further elaborated in Part V in relation to Actability Evaluation), and (b) desirable to design for good communication quality (Eriksson, 1998; 2002). The latter implies that communication through an IS should be comprehensible, should refer to the true state of affairs, should be performed in accordance with accepted social norms, and should reflect sincere intentions (Habermas, 1984). That is, from an IS design perspective, we are usually not interested in designing for hidden agendas; and even if we were, these agendas would be made known during design, and we could safely assume specific intended communication functions.

A further issue regarding multi-functionality, already pointed out in Chapter 4 and further elaborated in Chapter 7, is the fact that communication acts in general and communicative actions specifically are performed not only to accomplish changes in the social world, but are also performed so as to enact conformity with, for example, values, emotions and norms.

5.4 The Action-Elementary Message

The concept of the ae-message is explicitly derived from the notion of a multifunctional speech act of Figure 5-1. In accordance with the generic model of social action presented in Section 4.4.1 and the terminology introduced in Section 4.5.2 (for the five components of a communication act at page 73), the ae-message is understood
as the result\textsuperscript{32} of an e-action (which roughly corresponds to a speech act). As indicated above, there are different ways of conceiving the ae-message, of which two are seemingly equally intuitive. In the remainder of this section, these two views, and also a third, will be presented and the second will be argued as the one to be preferred.

5.4.1 A preliminary View of the Action-Elementary Message

One possible view of the e-action/ae-message constellation is that there is a one-to-one correspondence between these two: an e-action results in an ae-message and an ae-message is the result of one e-action. This view is depicted in Figure 5-2.

Taking its multi-functionality into account, this implies that the action mode of an ae-message may have several communication functions (illocutionary forces) embedded in it. However, as will be further elaborated in Chapter 13, this view on the concepts of the e-action and the ae-message have proved difficult to use as modelling primitives in real-life situations. The particular difficulty experienced relates to the question of how to identify ae-messages in the first place (once identified, they are usually readily analysed and documented). The main source of these problems seems to be the relation between the illocutionary component (the action mode) and the ae-message per se. Therefore we will now take a closer look at these concepts and eventually arrive at a slightly different view in Section 5.4.2.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure5-2.png}
\caption{The notion of an e-action resulting in an ae-message (Ågerfalk, 2002).}
\end{figure}

Based on the notion of a multi-functional speech act as depicted in Figure 5-1, Figure 5-2 shows the execution of an e-action by a communicator (speaker), which results in an ae-message. The execution is the physical performance of the e-action and might, for example, correspond to the pressing of a specific button in the user interface of an IS. The ae-message carries an action mode, which contains multiple (in this case three) communication functions (illocutionary components). All three are associated with the same propositional content. The first communication function is directed towards a specific interpreter (listener). The second communication function is directed towards another specific interpreter. This corresponds to the first case of multi-functionality described above – that is, one action that does two different things to two different interpreters. The third communication function is directed towards two differ-

\textsuperscript{32} Note that, as stated in Section 5.3, this ‘result’ should not be confused with the communication effects that the e-action has on interpreters.
ent interpreters, of whom one is also the subject for the second function. This corresponds to the second case of multi-functionality – that is, one interpreter is the subject of two communication functions.

Note that Figure 5-2 does not contain the ellipses of Figure 5-1. This is to indicate the design orientation inherent in the work on the ae-message concept. During the design of an IS, we, as designers, reckon with a specific number of intended communication functions and intended interpreters. Of course, an IS can be used in ways that were not intended from the outset, or can even be misunderstood, in which case the ellipses would apply. It is important, though, to distinguish such use of an IS from that intended in the original design.

Note that the concept of the ae-message should be understood as an abstract entity, representing the semantics and pragmatics of an e-action. Such a message must always have a corresponding physical representation (a medium). For example, ae-messages are typically received by interpreters in the form of graphical or textual representations on a screen document, which are more closely related to the syntax of the message (see Chapter 14). It is important to be aware that the structuring (layout) of documents has consequences for what we refer to as ae-messages in a given IS, since documents serve to hold together the various communication functions and propositional contents communicated: this structuring delimits, groups and explicates the message. Consequently, inappropriate mapping between ae-messages and documents may be a source of confusion. This would be the case, for example, if the propositional content expressed in a document did not match the intended communication functions (typically expressed by one or several illocutionary verbs), or if two ae-messages with different communicators and intended interpreters were mixed up in the same document.

Let us consider the trivial example of a person sending an e-mail message. The e-mail message could be seen as an ae-message, and sending it as an e-action. The recipients would probably agree that they all received the same message, even though they might each have interpreted it differently, and might very well act differently in response to it. This would accord with the view of the ae-message presented in Figure 5-2. From a computing perspective, this would also accord with the common view of e-mail messaging (the same message in terms of content, sender, time sent, and so on, would have been transmitted to multiple recipients). Obviously, this view of the ae-message seems rather intuitive.

Let us now turn to another, somewhat more intricate example. In most businesses, a typical communicative action is that of making a business offer. Table 5-1 shows the structure of a propositional content that may be used in such offers.

A propositional content of the type described in Table 5-1 consists of four e-messages referencing a product description (M1) and the price at which the product is offered, together with an expiry date (M2), and the names of the responsible salesperson (M3) and the customer (M4). To be complete in terms of Langefors’ theory, the example should also include a relational e-message referencing the other four, indicating that these combine to form an offer. By way of simplification, we can think of PC1 per se as embodying that relation. Note that the structure described by Table 5-1 should be understood as a type (or class) of propositional content of which actual instances are created during the performance of communication acts utilizing that type.
Table 5-1: A propositional content structure consisting of four e-message structures together describing what is talked about in a business offer (Ågerfalk, 2002).

<table>
<thead>
<tr>
<th>Propositional content</th>
<th>e-messages</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ID</td>
</tr>
<tr>
<td>PC₁</td>
<td>M₁</td>
</tr>
<tr>
<td></td>
<td>M₂</td>
</tr>
<tr>
<td></td>
<td>M₃</td>
</tr>
<tr>
<td></td>
<td>M₄</td>
</tr>
</tbody>
</table>

The ae-message Business Offer should be understood as a combination of Salesperson (the business role that communicators of the offer play), Propositional Content, and a combination of communication functions and intended interpreters (the function used with respect to particular interpreters). We can denote this structure as ae-message = (Communicator, Propositional Content, Performance Time, Communication Function × Intended Interpreter). If we assume that the e-action of making an offer also includes reporting to the sales manager that an offer has been made, we might end up with an ae-message structure as described in Table 5-2. Note that, as in the propositional content structure in Table 5-1, the ae-message structure in Table 5-2 should be understood as describing a type (or class) of ae-messages.

Table 5-2: A preliminary ae-message structure Business Offer (Ågerfalk, 2002).

<table>
<thead>
<tr>
<th>ae-message type</th>
<th>Communicator</th>
<th>Propositional content</th>
<th>Performance time</th>
<th>Communication function</th>
<th>Intended interpreter</th>
</tr>
</thead>
<tbody>
<tr>
<td>AE₁ Business Offer</td>
<td>Salesperson</td>
<td>PC₁</td>
<td>As of performance</td>
<td>Offer</td>
<td>Customer</td>
</tr>
</tbody>
</table>

In Table 5-2, the value PC₁ in the Propositional content column should be understood as a foreign key referencing the propositional content structure described in Table 5-1 – that is, PC₁ describes the propositional content structure used in e-messages of type AE₁. Note that the roles Salesperson and Customer appear both as (a) communicator and intended interpreter of ae-messages of the type described by the e-message structure in Table 5-2, and as (b) objects referenced by the associated propositional content described in Table 5-1. This is an important distinction, and should be understood to indicate that the salesperson utters something concerning himself and a customer, directed to that customer. To return to Austin’s example from Section 5.2, when uttering ‘I do’ the bridegroom both references himself and takes the bride as his lawfully wedded wife. Furthermore, he is also actually referencing the bride and the social institution of marriage, even though this is implied in the context (an implicit reference to the context that should usually be avoided in a formalized, computerized IS). The Performance time column is used to represent the time when the corresponding e-action is performed. This is an important dynamic property of ae-messages since the interpretation of ae-messages may not occur instantly as in face-to-face conversations (Auramäki et al., 1988; Clark, 1996) and positioning an action in its historical context may be vital for understanding.
Obviously, the communication of the business offer corresponds roughly to the e-mail example above. The offer could even be sent by e-mail to the customer, with a so-called carbon copy (cc) to the sales manager. Even though this notion of the ae-message seems intuitive, it is not the only option available.

5.4.2 An Elaborated View of the Action-Elementary Message

As indicated above, the ae-message plays a central role in the business and information modelling method presented in Part III of the dissertation. In deciding what information a system is required to maintain, and in designing the performance of e-actions at the user interface, ae-messages must be identified and described in detail. The resulting Message Definitions thus play a central role throughout systems development following the method. In order to identify ae-messages, one makes use of comprehensive descriptions of the business processes in which the IS under development is supposed to be used. The descriptions are depicted in so-called Action Diagrams. Among other things, Action Diagrams show activities (which consist of one or more e-actions) performed by human actors in interaction with information systems, together with information sets that are results of, and prerequisites for, activities. Figure 5-3 shows a minimal Action Diagram describing the business offer example introduced in the previous section. (In Action Diagrams, activities are shown within horizontal lines, and information sets are shown as inputs to, and outputs from, activities, which are represented by rhombuses connected to activities.)

![Action Diagram](image)

**Figure 5-3:** Action Diagram showing the making of a business offer, which includes reporting to management (Ågerfalk, 2002).

From the diagram in Figure 5-3, we can see that a Salesperson in interaction with an IS performs an activity, Make Offer, directed primarily towards a Customer, which results in a Business Offer. Additionally, an Offer Report is created for a Sales Manager for use in some further (unspecified) activity.

When working with Action Diagrams, it seems natural to distinguish the information set Business Offer from the information set Offer Report sent to management. Since this is not in line with the preliminary view of the ae-message presented above (which requires a distinction between information sets and ae-messages), confusion may arise when ae-messages are to be identified on the basis of Action Diagrams, and

---

33 The Action Diagram notation is explained in detail in Section 9.5.2, p. 158 ff.
practical experience shows that in fact it does. When describing business processes with Action Diagrams, the focus is on what actions are performed and what these actions do – how they shape the flow of activities constituting business processes. The main interest is in the pragmatic aspects of action, and consequently actors and illocutions are in the foreground, while information content (propositions) is seen mainly as the result of (and prerequisite for) action, as a subordinate part of action objects.

When moving on to analysis of ae-messages, the focus shifts towards the semantics of information, the issue of how information is structured. The propositional content is now analysed in terms of entities and their relationships, and is eventually associated with an action mode that describes how the information is used within the business, as derived from the Action Diagrams (see Table 5-1 and Table 5-2). What then should constitute a proper unit of analysis – that is, what is the ae-message?

When identifying action-elementary messages, taking the action as a starting point seems rather natural. Following the notion of a multi-functional speech act (see Figure 5-1) and adopting the terminology of Figure 5-2, the ae-message as described in Section 5.4.1 obviously takes the propositional content and the communicator as its elementary parts: it is a combination of a communicator, a propositional content and a combination of communication functions and interpreters. In this view, the ae-message is elementary with respect to what a communicator communicates about, not with respect to what the communicator does when communicating, which can hardly be considered action-elementary from a language/action perspective. This may also be a source of the problems associated with identifying ae-messages. If the basis for identifying ae-messages is the Action Diagram, which embodies a pragmatic perspective on information, and ae-messages are identified primarily on the basis of their semantics, then there needs to be a shift in perspective that is not only conceptually inelegant, but also potentially deceiving.

Based on this contention, the ae-message should be viewed as a ‘true’ combination of communicator, propositional content, communication function and intended interpreter, which can be denoted as ae-message = (Communicator, Propositional Content, Performance Time, Communication Function, Intended Interpreter). In this view, the e-action of making an offer as depicted in Figure 5-3 should be understood as producing two separate ae-messages, one for each intended interpreter. Table 5-3 shows this view of the ae-message.

**Table 5-3:** Elaborated ae-message structures Business Offer and Offer Report (Ågerfalk, 2002).

<table>
<thead>
<tr>
<th>ae-message type</th>
<th>Communicator</th>
<th>Propositional content</th>
<th>Performance time</th>
<th>Communication function</th>
<th>Intended interpreter</th>
</tr>
</thead>
<tbody>
<tr>
<td>AE1 Business Offer</td>
<td>Salesperson</td>
<td>PC1</td>
<td>As of performance</td>
<td>Offer</td>
<td>Customer</td>
</tr>
<tr>
<td>AE2 Offer Report</td>
<td>Salesperson</td>
<td>PC1</td>
<td>As of performance</td>
<td>Report</td>
<td>Sales manager</td>
</tr>
</tbody>
</table>

In this view, the ae-message is elementary with respect to a communicator who does something to an interpreter, by saying something about something. This view, depicted in Figure 5-4, clearly accords with the original notion of the multi-functional speech act as presented above. In this view, a communicator executes an e-action,
which results in several intended ae-messages, each directed towards one or several intended interpreters.

![Diagram](image)

**Figure 5-4:** The notion of an e-action resulting in several (elaborated) ae-messages (Ågerfalk, 2002).

In practical systems work, this means that identifying ae-messages can be based on the so-called illocutionary verb used to talk about actions in the business, for example, ‘offer’ and ‘report’, as is done when working with Action Diagrams; there is no need for a potentially deceiving shift in perspective.

A further, also practically grounded advantage of viewing the ae-message this way is that the unit of analysis in systems work becomes more modularized, which may potentially lead to reuse of specifications, or at least less reworking. This can be exemplified in a re-examination of the offering action discussed above. Let us assume that the action, Make Offer, eventually turned out to be actually an aggregate of two actions performed in sequence, or that systems developers decided to provide such a design as an alternative. See Figure 5-5 for an illustration. (The dotted line in the diagram shows a temporal dependency between the two activities.)

![Diagram](image)

**Figure 5-5:** Action Diagram showing the making of a business offer, with reporting to management as a separate activity (Ågerfalk, 2002).

This would be the case, for example, if the report action were performed by another user interface manipulation (execution), maybe through another system or simply at a later point in time. In this case, the two messages (Business Offer and Offer Re-
port) could hardly be considered as the same ae-message. Therefore, given that initially they were, this re-design would require a reworking of the original documentation. Viewing the ae-message in the way proposed in this section, such a re-design would not require any reconsideration of the messages sent through the system, only of the way in which they are being used.

The proposed view of the ae-message has further consequences for documenting and analysing ae-messages during systems development. Obviously, communicating messages that are identical in every respect, except for communicative function in relation to a range of different interpreters, potentially requires the description of a huge number of similar ae-messages. This can, however, be handled pragmatically by treating them as variants of the same message. Thus, the description of ae-messages would look similar to that of Table 5-2. One should not confuse this description with the concept of the ae-message per se.

5.4.3 A Rejected View of the Action-Elementary Message
A further alternative view of the ae-message would be to say that one execution (i.e. utterance act) results in several e-actions. This would introduce a distinction between the physical utterance (for example, pressing a particular button in the user interface of an IS) and the social action thus performed. This would imply, for example, that sending an order by clicking a button labelled ‘Send Order’, and sending the same order by clicking a menu item with the same caption, would result in different ae-messages. Such a solution would be neither intuitive nor consistent with speech act theory in general, and has therefore been rejected.

5.5 Introducing the IT Artefact
In the treatment of the ae-message above, it has to a large extent been discussed as would it appear in an ordinary face-to-face conversation; the setting usually assumed by speech act theory. It is commonly believed that speech act theory concepts are equally valid in written settings as in a spoken settings (Searle, 1969; Graham, 1998). As pointed out by Clark (1996), the written setting does, even though similarities are obvious, deviate significantly from a spoken setting. Introducing an IT-based information system puts even further restrictions upon the use of language (Eriksson, 2000; 2002; Schoop and Quix, 2001). In this section we will explore how such a setting influences our conception of the ae-message. A basic assumption is that the performance of business actions through and by means of information systems can be characterized as what Clark (1996) terms a ‘prescriptive setting’ or at least an ‘institutional setting’. In an institutional setting, the participants’ engagement in speech exchanges resembles an ordinary private conversation, but is limited by institutional rules. Business conversations are institutional since they conform to certain business rules and norms. A prescriptive setting is even more restricted in that the words spoken are largely (or completely) fixed beforehand. This is usually the case with business information systems since they implement a professional language that is enforced on users (Goldkuhl and Lyytinen, 1982). This condition is manifested in the systems as the set of possible e-actions to perform and the corresponding ae-messages possible to create – the action potential of the system.
5.5.1 Participants in the Conversation

Section 4.5.1 introduced three different meta-roles of IS users: the communicator, the performer and the interpreter. A performer, being the one performing an action interactively directed towards an intended interpreter, may perform his action on behalf of another actor, referred to as the communicator. So far in the treatment of the ae-message no distinction has been made between communicators and performers; by way of simplification they have been treated collectively as communicators. By incorporating these meta-roles, we arrive at a further elaborated concept of the ae-message, as depicted in Table 5-4.

<table>
<thead>
<tr>
<th>ae-message type</th>
<th>Communicator</th>
<th>Performer</th>
<th>Propositional content</th>
<th>Performance time</th>
<th>Communication function</th>
<th>Intended interpreter</th>
</tr>
</thead>
<tbody>
<tr>
<td>AE$_1$ Business Offer</td>
<td>Sales dept.</td>
<td>Salesperson</td>
<td>PC$_1$</td>
<td>As of performance</td>
<td>Offer</td>
<td>Customer</td>
</tr>
<tr>
<td>AE$_2$ Offer Report</td>
<td>Salesperson</td>
<td>Salesperson</td>
<td>PC$_1$</td>
<td>As of performance</td>
<td>Report</td>
<td>Sales manager</td>
</tr>
</tbody>
</table>

From Table 5-4 we can see how the salesperson communicates the offer to the customer on behalf of the sales department. With respect to the offer report sent to management, on the other hand, the salesperson is both communicator and performer. Altogether this means that even though the salesperson performs both actions, it is the sales department that is responsible for the offer in relation to the customer, while it is the salesperson that guarantees the correctness of the offer report in relation to the sales manager.

5.5.2 Input Messages and Output Messages

In addition to the three meta-roles discussed above, a distinction between input messages and output messages was introduced in Chapter 4 (see Figure 4-3, p. 63). An input message is the result of an interactive e-action performed at the user interface while an output message is an ae-message for interpreters to interpret. What then is the relation between ae-messages that are input to an IS and those that are output? First of all, we must remember that all ae-messages share the property of having a communicator, a performer and an intended interpreter. If the performer is also the communicator and what comes out of the system is identical to what goes in, such as in the e-mail messaging case, we are obviously dealing with the same ae-message as both input and output. Section 4.5.4 introduced the notion of the IS as a communication mediator (see Figure 4-8, p. 75). This means that there might be output messages that are derived from earlier input messages, that is, output messages that are results of automatic information processing (such as aggregation and summation) involving several other ae-messages as a basis. Understanding the propositional content of such a message and how it relates to previous messages is straightforward and can be achieved through traditional conceptual data modelling. But, if we want to maintain the social action perspective inherent in the concept of the ae-message: Who, then, would be the communicator, the performer and the intended interpreter of such a message? In order to answer that question, we need to consider the view of information systems as performers of action as discussed in Chapter 4.
Let us therefore continue the Business Offer example and consider what happens when the customer eventually accepts the offer and decides to communicate an order in response. The order action by the customer may then result in an ae-message according to the structure depicted in Table 5-5.

**Table 5-5: The ae-message structure Order.**

<table>
<thead>
<tr>
<th>ae-message type</th>
<th>Communicator</th>
<th>Performer</th>
<th>Propositional content</th>
<th>Performance time</th>
<th>Communication function</th>
<th>Intended interpreter</th>
</tr>
</thead>
<tbody>
<tr>
<td>AE3 Order</td>
<td>Customer</td>
<td>Customer</td>
<td>PC2</td>
<td>As of performance</td>
<td>Order</td>
<td>Order receiver</td>
</tr>
</tbody>
</table>

For the example, we assume that the communicator and the performer of the Order message is the customer and that the propositional content of the Order message (PC2) references the previously communicated Business Offer (AE1) and some extra information (an order date, for example). Let us further assume that the communication function is an order and that the intended interpreter is an Order receiver in the Sales Department. This ae-message structure follows the pattern of AE1 and AE2 and is an example of a message that goes straight through the system; the output message corresponds to the input message.

If we also assume that the sales manager is interested not only in getting business offer reports, but wants also to be updated on the sales activities, an additional ae-message Order Report is required. Who then would be the communicator of such a report? Since we cannot possibly make the customer responsible for reporting to management, it must be the order receiver that is the communicator of the Order Report. Such reporting does not have to be performed as a separate activity by the order receiver, but can be performed automatically whenever an order is received. In that case, the order receiver (or the sales department, depending on the actual business design) is the communicator of the Order Report message while the IS itself is the performer – an automatic action. Messages resulting from automatic actions, just as any ae-message, always have a communicator, even though the performer is an IS.

### 5.5.2.1 Derived Messages and Automatic Action

Suppose now that the sales manager does not want to view a report for every single order posted but instead wants to have access to a complete order backlog. How would we treat that situation in terms of ae-messages? The required Order Backlog message (an output message) would then be derived (in this case aggregated) from a set of Order Report messages (input messages). Since the sales manager is the intended interpreter of the Order Report messages, this situation can be treated as would the sales manager both communicate and intend to interpret the Order Backlog. The communicator as well as the intended interpreter of Order Backlog messages would thus be the Sales manager, whilst the performer would be the IS. Note that the Performance time of Order Backlog messages would deviate from the Performance times of the single Order Report messages and correspond to the time the backlog was calculated for the manager. Table 5-6 depicts the resulting ae-message structures, where PC3 refers to an Order message (AE3) and PC3 refers to the aggregated information derived from several AE4 messages.
Table 5-6: The ae-message structures Order Report and Order Backlog.

<table>
<thead>
<tr>
<th>ae-message type</th>
<th>Communicator</th>
<th>Performer</th>
<th>Propositional content</th>
<th>Performance time</th>
<th>Communication function</th>
<th>Intended interpreter</th>
</tr>
</thead>
<tbody>
<tr>
<td>AE4 Order Report</td>
<td>Order receiver</td>
<td>IS</td>
<td>PC3</td>
<td>As of performance</td>
<td>Report</td>
<td>Sales mgr</td>
</tr>
<tr>
<td>AE5 Order Backlog</td>
<td>Sales mgr</td>
<td>IS</td>
<td>PC3</td>
<td>As of performance</td>
<td>Report</td>
<td>Sales mgr</td>
</tr>
</tbody>
</table>

Altogether this means that any output message corresponds to an input message, or is the result of an automatic action. Output messages that are results of automatic action may or may not be derived from other previous messages stored in the IS. This also implies that there may be input messages that do not correspond directly to any output messages. Such messages are instead used as a basis for automatic actions resulting in output messages. This would be the case with AE4 messages, which are not used directly but serve as a basis for the creation of AE5 messages. The intended interpreter of AE4 messages is the sales manager. The sales manager, however, has delegated the reception and further refinement of such messages to the IS. Based on this we can also define ‘automatic action’ more precisely to mean the automatic production of ae-messages that do not correspond to any input ae-messages.

This way of viewing automatic information processing as delegation of social communicative responsibility has the advantage of making it possible to pinpoint exactly who is responsible for actions performed through a system. Even though a system owner may be responsible for the correct processing of messages, such as the mapping between AE3 and AE4 messages, it is the actors delegating actions to the system that are responsible for the social obligations created during system use, as they are for any other social activity in which they participate. In our example, this means that it is the responsibility of the sales manager to uphold quality in the order backlog procedures involved in Order Backlog messages. It also means that it is the order receiver that is responsible for the Order Report. This latter condition implies the importance of making order receivers fully aware of the fact that an order report is being created on their behalf, possibly at the time of order reception. Thus promoting ‘white boxing’ of the IS and avoiding ‘automagic’ functionality for which no one feels responsible.

5.5.2.2 Mapping Messages to Screen Documents

As discussed above, messages are often carried and visualized by screen documents (forms, dialogs, windows, et cetera). It seems that information shown in a particular screen document often corresponds to a number of ae-messages. For example, when working with the actability evaluation method (in Application 7, see Part V of the dissertation) we found a screen document used to book resources using a corporate intranet that showed, among other things, information about the name of the particular resource, information about current bookings, and information about the status of the booking (whether it is a booking request or a confirmed booking). These ‘information fragments’ that are needed to get a complete picture of the situation in order to do a new booking are all derived from different ae-messages. The name of the resource is derived from the declaration of the resource as an item possible to book, made by a resource manager directed towards people performing bookings – the ‘bookers’ (see ae-message AE6 in Table 5-7). The current booking information is derived from a previous booking action (a request) made by a Booker directed towards the person respon-
Messages in information systems

able for the resource (the resource owner) and towards other bookers as a report. This type of e-action thus gives rise to two different types of ae-messages, AE7 and AE8 (see Table 5-7). The status information is derived from a confirmation action made by the resource owner directed towards the particular requesting booker (as an answer to the request) and towards other bookers (as a report that the resource is now booked). Thus, this type of e-action also gives rise to two different types of ae-messages, AE9 and AE10 (see Table 5-7).

Table 5-7: Example ea-message structures used in a system for booking resources.

<table>
<thead>
<tr>
<th>ae-message type</th>
<th>Communicator</th>
<th>Performer</th>
<th>Propositional content</th>
<th>Performance time</th>
<th>Communication function</th>
<th>Intended interpreter</th>
</tr>
</thead>
<tbody>
<tr>
<td>AE6 Declare Resource</td>
<td>Resource Manager</td>
<td>Resource Manager</td>
<td>PC5</td>
<td>As of performance</td>
<td>Declaration</td>
<td>Booker</td>
</tr>
<tr>
<td>AE7 Request Resource</td>
<td>Booker</td>
<td>Booker</td>
<td>PC6</td>
<td>As of performance</td>
<td>Request</td>
<td>Resource owner</td>
</tr>
<tr>
<td>AE8 Report Request</td>
<td>Booker</td>
<td>Booker</td>
<td>PC6</td>
<td>As of performance</td>
<td>Report</td>
<td>Booker</td>
</tr>
<tr>
<td>AE9 Acknowledge Booking</td>
<td>Resource owner</td>
<td>Resource owner</td>
<td>PC7</td>
<td>As of performance</td>
<td>Answer to Request (Requesting)</td>
<td>Booker</td>
</tr>
<tr>
<td>AE10 Report Booking</td>
<td>Resource owner</td>
<td>Resource owner</td>
<td>PC7</td>
<td>As of performance</td>
<td>Report (Other) Booker(s)</td>
<td></td>
</tr>
</tbody>
</table>

The screen document thus shows information derived from the ae-messages in Table 5-7 (and from some additional that has been left out to simplify the example). However, the screen document does not show all information contained in these messages. This is important to note because it is important that information shown is faithful to its origin. That is, a person communicating a message probably assumes that the content of that message is communicated to intended interpreters in its entirety. Therefore, when designing documents it is important to trace the origin of messages displayed and to ensure that a sufficient amount of the propositional content from each message is displayed, so that the context is not ignored.

On the other hand, since most interactive screen documents are used also to perform e-actions, it is important to distinguish the output messages shown from the possible input messages to create and send. Certainly, not all the information shown in a document is used in every possible action to perform using that document. In Table 5-7, this is evident since a booking request (AE7) contains less information than what is shown in the screen document, that is, the sum of AE6 + AE7 + … + AE10.

5.6 Messages in Information Systems – A Summary

In this chapter, the argument for a shift from a descriptive perspective on information and information systems to a view of information as action and communication, which was introduced in Chapter 4 has been extended. This was done on the basis of the work
of Goldkuhl (1995) and by way of clarifying the concept proposed in Chapter 4 of an action-elementary message (ae-message). On this basis, an ae-message should be conceived as containing an action mode that consists of one, and only one, communication function in relation to its intended interpreter(s). This communication function should in itself be regarded as multi-functional (or at least multi-dimensional), since it may create obligations on the part of the speaker, and since it is both expressive and appellative. The reconsidered ae-message can be described as a combination of communicator, performer, propositional content, performance time, communication function and interpreter, and is truly elementary with respect to all its components. It is thus an elementary unit of information as action and communication that suggests a foundation for an integrated approach to conceptual modelling and business modelling based on speech act theory. It also promotes a ‘white box’ approach to information systems that facilitates a move from focusing system ownership towards focusing action responsibility.
Chapter 6

Pragmatization of Conceptual Modelling


6.1 Introduction

Conceptual modelling, as a systems development activity, is basically concerned with two different views: a **static** and a **dynamic** view. The static view emphasizes static properties in terms of entities and relationships (Chen, 1976). The dynamic view captures how entities in the static model change state over time. Such state changes are thought of as triggered by events occurring in the system’s environment (e.g. Booch et al., 1999).

Although conceptual modelling is an important activity in the systems development process, there is confusion as to how it should actually be performed. There is, for example, no consensus on how to represent associations in the static model (such as weak entities, relationships and attributes) (Wand et al., 1999). There are also problems related to the modelling of dynamic and temporal aspects (Gregersen and Jensen, 1999; Snoeck and Dedene, 1998). A further problem concerns pragmatic aspects, which are largely neglected in traditional conceptual modelling. This implies that pragmatic concepts, such as actors, responsibilities, actions and commitments, are not paid sufficient attention during conceptual modelling (Goldkuhl, 1995; Nurminen, 1988).

These circumstances have consequences for the information system under development. They may, for example, imply that the system fails to provide relevant information to users, that users do not understand how to use it (e.g. Gulliksen et al., 1997), and that users cannot trace who is responsible for information, actions and commitments made (Erickson and Kellogg, 2000; Eriksen, 2002).

One important cause of these problems is that traditional conceptual modelling is based on a descriptive perspective on information systems, which embodies an objectivist view of reality (Holm, 1996).

This chapter contains a discussion of how conceptual modelling can be informed by an action-oriented perspective of information systems to provide a practical, yet theoretically well-founded, basis for capturing important properties of the business modelled. Properties such as actors, actions, associations, responsibilities, time constraints and state-changes. The pragmatic aspect is here seen as related to how the system is used to perform actions not only on the basis of information from the system but also through the system, as discussed in Chapters 4 and 5.

An integration of traditional conceptual modelling with action-oriented business modelling is proposed, an integration that may eliminate the tendency of contemporary
conceptual modelling approaches to yield overly abstract models that lack a clear path from model to implementation in business information systems. One, perhaps even the foremost reason why conceptual models tend to be too abstract is that the use of conceptual models and information systems is not analysed pragmatically from within the business context. Instead, conceptual models and information systems are considered as images or mirrors that are external to the business.

6.2 Beyond the Descriptive Perspective

When creating a conceptual model, from a descriptive perspective on information systems, the business at hand constitutes the universe of discourse, i.e. the part of reality that the model claims to reflect. The model is then transformed into a computational representation and stored in the database of the system being developed. (This is admittedly a somewhat simplified description, but it is sufficient for the purpose of this chapter). This model of the business is then used as a source of knowledge about the business, i.e. the business actors can use the computerized model instead of looking directly at the world (see Figure 6-1). This objectivistic view of information systems and of reality reduces the user to an observer who is observing the real world from the outside, through the conceptual model as implemented in the system (Lyytinen, 1987b). As discussed in Chapters 4 and 5, the pragmatic aspect is here seen as mainly related to how the user acts on the basis of information acquired from the system; i.e. action is conceived as being external to the system.

![Figure 6-1: Traditional view of an information system as an image of reality (Ågerfalk and Eriksson, 2002).](image)

As discussed above (Chapters 4 and 5), the descriptive view of information and information systems as an image of reality has been challenged and criticized within the language/action perspective. From a language/action perspective, an information system is regarded not as an image of reality that stores true information about the world but rather as a vehicle for social action and communication within a business context (see Chapter 4).

However, conceptual modelling is still important within such an action-oriented view of information systems. One reason is that the systems must contain and provide a business vocabulary that includes concepts used for communication (Goldkuhl and Lyytinen, 1982; Lyytinen, 1986). Furthermore, the systems must store information about the current state of the business and maintain a record of business actions per-
formed. In fact, as we shall see in the next section, applying an action-oriented view of information systems implies a reconciliation of ‘traditional’ conceptual modelling and the pragmatic aspects of language and computer use.

With the notion of the conceptual model (and the information system) as an image of reality, the predominant modelling problem is to analyse how the external reality should be mapped into, and represented in, the system in a ‘true’ way. In contrast to this, the main modelling problem should be to analyse the communication acts performed by use of the system within the business, and how these acts may affect the business context. With such a pragmatic view, conceptual modelling is still important, but the importance of the business context must be emphasized more. In the next section, speech act theory will be used to give a theoretical background for such a pragmatic approach to conceptual modelling.

6.3 Information Systems and Speech Acts

The assumption that information systems should serve as images of reality, used to inform users about the world, suggests a rather restrictive view of the role of information systems in business. Information systems are, on the contrary, primarily used for communication (Flores, 1998) and this activity cannot be viewed only as making descriptions of reality. People do not use language only to talk about events in the external world as observers; they act and communicate within the world, as social actors. Communicating implies doing things, and messages carry more meaning than just facts about reality: they also carry the actors’ intentions and beliefs and are used to influence people and to change the world. Communication can therefore be viewed as action. This is also the main idea in the theory of speech acts (Austin, 1962; Searle, 1969). According to Austin (1962), ‘to speak is to act’. When saying something, we are doing something – for example, promising or commanding. As discussed in Section 1.3.3, Austin also coined the phrase ‘descriptive fallacy’ to refer to the misconception that language is only used for descriptions of reality; which is the case with a traditional conceptual view of information and information systems.

In Chapter 5, the concept of the e-action (elementary action) was described as a multi-functional speech act consisting of an execution (utterance act) that results in one or several ae-messages (action-elementary messages). Each of these ae-messages references the same propositional content (propositional act) and has a specific communication function (illocutionary act) directed to one or several interpreters. An e-action also has certain indented communication effects (perlocutionary acts) aligned with the communication function. It is important to emphasize that there are a number of effects that can be the result of performing an e-action, and that these effects can also be oriented towards the speaker. For example, the effect of a promise is the creation of a commitment on the part of the speaker, to perform a subsequent act (see Section 5.3.2).

All semiotic acts, both linguistic and non-linguistic, must be understood within the social context in which they are uttered (Clark, 1996). Actors must understand the context to participate successfully in communication. Auramäki et al. (1988) define the context of a speech act to be a combination of speaker, hearer, time, place and possible world. The first two concepts refer to the actors who are performing and interpreting the action. Time and place represent the temporal and spatial aspects of the action. Possible world refers to the residual features of the context that make a particular action possible and meaningful, and hence potentially successful. Typically, these include shared norms, values and beliefs and the existence of certain social and material (brute)
facts. Note that by referring to a possible world rather than the actual world, it becomes possible to talk about the future and what ought to be (Aurumäki et al., 1988). When doing business, the social context of the communication is a **business context**. From this discussion, it is possible to conclude that the business context consists of actors, situated in time and space, performing communicative and material actions, and these actions must be related to an inter-subjective understanding of the business context.

This theoretical discussion can be applied to the use of an information system in a business context with the help of the business offer described in Figure 6-2. The business offer is communicated by the use of a sales support system within a car sale/purchase business context (Eriksson, 1998). The business offer above can be understood as an ae-message, which consists of a propositional content and an illocutionary component (communication function). In the message, the propositional content identifies and describes the attributes of the purchase object, which is a car. The illocutionary component shows how the propositional content should be used; in this case it should be understood and used as a business offer.

<table>
<thead>
<tr>
<th><strong>Business Offer</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Date:</strong> 18-12-97 <strong>Time:</strong> 10:30am</td>
</tr>
<tr>
<td><strong>Salesman:</strong> James Howard</td>
</tr>
<tr>
<td><strong>Customer:</strong> Jenny Doe</td>
</tr>
<tr>
<td><strong>Model:</strong> Volvo 850 GLT 2.5, Front-wheel drive, 4 doors.</td>
</tr>
<tr>
<td><strong>Engine:</strong> Petrol. Catalytic. 5-cylinder. Transverse. 170 h.p./125 kW B5254F</td>
</tr>
<tr>
<td><strong>Gear Box:</strong> Manual 5-ganged. M56.</td>
</tr>
<tr>
<td><strong>Color:</strong> Polar white nr: 189</td>
</tr>
<tr>
<td><strong>Extra Equipment:</strong> S-Package: Automatic climate control + Cruise control.</td>
</tr>
<tr>
<td><strong>Amount due:</strong> £16,000</td>
</tr>
<tr>
<td><strong>Offer valid until 31-12-97.</strong></td>
</tr>
</tbody>
</table>

**Figure 6-2:** A Business offer (Ågerfalk and Eriksson, 2002).

If relating this example to Searle’s description of different sub-acts and effects (see Section 5.3.1), it can be described as follows:

a) The **utterance act** is the production and communication of the physical written message, which can be presented on the computer screen or printed on paper.

b) The **propositional act** is performed through the propositional content, which refers to an object, in this case a purchase object (a car) and its attributes.

c) The **illocutionary act** is performed through the illocutionary verb *offer*.

d) The business offer affects the business context; for example, by creating an obligation on the part of the car dealer to sell the car under the conditions described in the offer. The offer may also create the effect of the subsequent purchase of the car by the customer, which is likely the intention of the car dealer.

The business context of the communication act consists of: time (18-12-97, 10:30 am), place (car dealer’s office), speaker (car dealer together with the system), hearer (customer), and possible world (the purchase object, i.e. the car and the price of the car,
as described by the propositional content; and business rules, social expectations and beliefs that govern the actors’ behaviour).

When the communication act is performed, it changes the state of the business context to the Car Offered state. The transition implies that (a) information, which consists of the propositional content, has been created, (b) the car dealer has expressed the intention and will to sell the car and (c) a commitment has been created on the part of the car dealer to sell the car under the conditions that are described in the offer.

If we agree that communication is action that involves causes and effects that change the state of the business context, and that these actions can be performed by the use of information systems, then we can use Searle’s theoretical discussion to understand the integration of the conceptual aspect of information systems with its pragmatic aspects. However, let us first give some examples of how speech act theory has been used as an alternative to the strict representational view of information systems in the information systems development community.

### 6.4 Conversation for Action

Traditional descriptive conceptual modelling focuses on the propositional content of information. The idea is to find generic information structures that are stable over time. However, detaching the propositional content from its intended use is a prominent example of Austin’s descriptive fallacy (Winograd and Flores, 1987; Holm, 1996).

To put more focus on the illocutionary component that is neglected by conceptual modelling, Winograd and Flores (1987) propose a modelling approach based on speech act theory and the coordinating power of language. Their ‘conversation-for-action’ (CFA) schema (see Figure 6-3), which is based on transitions between states, has had a substantial impact on our understanding of computer supported collaborative work and has been of great importance for workflow management.

![Figure 6-3: The Conversation for Action (CFA) schema (Winograd and Flores, 1987).](image-url)

Winograd and Flores (1987) explain that the CFA schema is derived from the observations that computers can be used to support human communication and that computers should be programmed on the basis of repetition and reoccurrence. Their conclusion is that to design information systems that support human communication, devel-
opers must identify repetitive and reoccurring structures. They further claim that the CFA schema captures these structures accurately at a generic level.

According to the CFA schema, a business conversation is initiated by a request from a customer (the initial speaker), which specifies some conditions of satisfaction. The supplier (the initial hearer) then has the choice of accepting the conditions (promising to satisfy the request), rejecting it or making a counteroffer. If and when the parties have agreed, the supplier eventually asserts that the conditions of satisfaction have been met. The customer can then either declare that, in his or her opinion, the conditions have not been met, or express satisfaction, thus ending the conversation happily. During the conversation, both the supplier and the customer can withdraw at any point and thus cancel the conversation sequence unhappily.

Building on the generic speech act pattern of the CFA schema, the Action Workflow approach (Denning and Medina-Mora, 1995) describes business interaction as consisting of four phases: (1) preparation, (2) negotiation, (3) performance and (4) acceptance. As before, the roles of customer and performer are pre-defined. These phases and roles are described by the Action Workflow loop (see Figure 6-4).

![Figure 6-4: The Action Workflow loop (Denning and Medina-Mora, 1995).](image)

Both the CFA schema and the Action Workflow loop can thus be regarded as generic schema for the structure of business activity used to direct analysts’ attention to the action-oriented character of doing business.

The advantage of these state-transition modelling techniques is that they take into consideration the illocutionary component and the way that speech acts affect the business context, when they model the business to design information systems. However, a problem with these approaches, and similar ones such as DEMO (Dietz, 2001) and COMMODIOUS (Holm and Ljungberg, 1996), is that they not only shift from a narrow focus on the propositional content of information, but that they actually tend to disregard the important coupling between the propositional content on one hand and the illocutionary component and the way that speech acts affects the business context on the other. The speech act based modelling techniques described above become as narrow as the methods used for traditional conceptual modelling but with another focus.

In the next section, an alternative to contemporary speech act based approaches is discussed, an alternative that considers both propositional contents and illocutionary components and how speech acts affect the business context.

### 6.5 Action-Oriented Conceptual Modelling

Taking communication acts performed in a business context as the starting point for systems development, as suggested in Section 6.3, implies that business design and
information systems design become integrated into a single activity that encompasses both of them. This is so because information systems are not used only for storing and providing information about an external reality (i.e. the business at hand), which is the view of information systems in traditional conceptual modelling. It is important to understand the system as a vehicle used for performing communicative business actions embedded in a business context, which the system also affects.

From the discussion in Section 6.3, we have seen that communication can be seen as action and that the performance of communication acts is a presupposition for subsequent communication acts as well as for other ‘non-linguistic’ acts (i.e. communication acts may trigger subsequent acts). This means that business modelling and systems development become an integrated activity of analysing the whole context of action. With the previous discussion in mind (see Section 6.3), let us take a look at three important aspects of communication acts and conceptual modelling. First, the propositional content of a communication act can be described by traditional static conceptual modelling (such as E/R modelling or object modelling). Second, the illocutionary component of the communication act should be analysed together with the propositional content; this is important for both static and dynamic modelling. Finally, actions change the state of the business context and thus ought to be the basis of dynamic conceptual modelling. These observations can be related to the distinction that is made in linguistics between the semantics and pragmatics of language. Semantics is focused on the meaning of propositional content while pragmatics is focused both on the pragmatic meaning of language when it is used in communication situations and the effects that the communication have on speakers and hearers. Based on this discussion, it is possible to claim that:

- traditional conceptual modelling has focused too much on the semantics of language and too little on the pragmatic aspects (see Section 6.2); and that
- speech act based modelling techniques have focused too much on the pragmatics of language and too little on the semantic aspects (see Section 6.4).

However, by using an action-modelling approach, we can take both the semantic and pragmatic aspects into consideration. This approach makes it possible to reconcile traditional conceptual modelling and the pragmatic aspects of language and computer use. In the remainder of this section, an example of how this can be done is given. The example is derived from a case study (Eriksson, 1998) of a sales support system used in a car sale/purchase business context. It is important to note that the example is not intended to provide empirical evidence; rather, it should be understood as an illustration of the discussion based on a real-world example.

**6.5.1 Dynamic Modelling**

The main idea of an action-oriented perspective is to analyse the way that information systems are used to perform actions within a business context. Using Action Diagrams is one way of doing this (see Section 9.5.2). The Action Diagram in Figure 6-5 shows five important business actions (offer, purchase, order confirmation, delivery and payment) that are performed during the car sale/purchase business process. Action Diagrams show actions and performers of actions together with information and material flows that are results of, and preconditions for, actions. The Action Diagram of Figure 6-5 also utilizes an additional feature, tagged boxes, showing how the actions change the state of the business process. This way, state transitions and actor-communication
links can be modelled together. Let us now examine further the meaning of the actions and the state transitions described in Figure 6-5.

**Figure 6-5:** Action Diagram describing the sale/purchase process (Ågerfalk and Eriksson, 2002).

**Action 1 – Offer:** The car dealer, together with the sales support system, performs the business offer action, which is a communication act. The transition to the Car Offered state means that the car dealer has expressed the intention to sell the car. It also implies that a commitment has been created on the part of the car company to sell the car under the conditions described by the propositional content of the business offer; for example, not to sell the car at a price higher than that stated in the business offer.

**Action 2 – Purchase:** The customer makes use of the business offer to decide whether to buy the car or not. If the customer decides to perform the purchase action, using the sales support system, then there is a transition to the Car Ordered state. The Car Ordered state means that the customer has expressed the wish that the car dealer should sell the car to him or her. It also implies that a commitment has been created on the part of the customer to pay the price and to buy the car under the conditions that are described by the propositional content of the purchase order.
Action 3 – Confirmation: The order confirmation action, which also is a communication act, is performed by the car dealer, together with the customer, with the help of the sales support system, and the order confirmation is manifested in a purchase contract. The Order Confirmed state means that the car dealer has expressed willingness to accept the purchase order from the customer. It also means that the customer has confirmed their intention of buying the car. New commitments are also created. One commitment implies that the car dealer will deliver the car under the conditions described in the purchase contract. Another commitment is created on the part of the customer, implying that he or she will pay for the car under the conditions specified in the business contract.

Action 4 – Delivery: The supplier performs the delivery action, which is a material act. The Car Delivered state implies that the car dealer (or rather the car company) has fulfilled the commitment to deliver the car under the conditions that were specified in the business contract.

Action 5 – Payment: The customer performs the payment action, which can be either a material act or a communication act depending on how the payment is carried out. The Payment Issued state implies that the customer has fulfilled the commitment to pay for the car under the conditions that were specified in the business contract.

It is important to do action modelling because it shows that actions are performed with the help of information systems. These actions create information and change the state of the business process by creating commitments that must be considered and fulfilled as it proceeds.

In traditional conceptual modelling, the system is not viewed as a vehicle for performing actions embedded in a business context. With a descriptive perspective, all actions are performed outside the system, which is only used for description of an external world constituted by actions, objects and events. Certainly, information systems are used for describing actions and events that are external to the system. For example, the sales support system could be used to describe the material act of delivering the car. A clerk at the delivery office who registered a delivery report in the system would typically do this. Although this message is used to describe an external action (event) in relation to the system, it is important to emphasize that the very registration of the delivery of the car (and hence of the issuing of the delivery report) should be seen as a communication act. The Report action, which is performed by use of the system, states that the delivery action has taken place. It is important to emphasize that the clerk at the delivery office who registers the information is not only making a description of the delivery of the car (which is the part of the communication act that is called the propositional act, see Section 6.3), but is also stating that this is a fact (which is the part of the communication act that is called the illocutionary act, see Section 6.3), which commits the clerk to the truth of the propositional content. This Report action will of course not change the physical delivery of the car; nevertheless, it affects the business context because it implies that the car company claims that it has fulfilled its commitment. The state of the business is changed to Delivery Confirmed, which implies (a) the creation of information about the delivery of the car, (b) the making of a commitment on the part of the clerk regarding the truth of the report and (c) a declaration that the car company has fulfilled its commitments, which implies that it is free to invoice the customer, i.e. to request that the customer should fulfil the commitment to pay for the car.

The Action Diagram in Figure 6-5 and the description of the business process are, of course, simplifications of the activities performed in the actual business process. For
example, the negotiation that takes place in the Proposal phase of the car deal has not been described. Normally, there is a negotiation between the car dealer and the customer when the car dealer has made the initial business offer. The purpose of the Action Diagram has been to indicate a number of generic actions in the business process that can be used to illustrate the points to be made, i.e. to illustrate that:

Â the business context and the way that the system is used within this context is the focus, rather than a universe of discourse that is external to the system; that
Â the business context is constituted of both material acts and communication acts, and both action types must be analysed when we model the business, to analyse how the system is used in, and affects, the business context; that
Â the actions performed (both material acts and communication acts) change the state of the business context; and that
Â the communication actions performed are carried out together with the IS.

6.5.2 Static Modelling

The static view of the business emphasizes static properties of entities and their static dependencies. Static modelling has been the focus of traditional conceptual modelling techniques but so far has not been sufficiently included in speech act based modelling techniques such as Action Workflow (Denning and Medina-Mora, 1995) and DEMO (Dietz, 1994; 2001). It is important from an action-oriented perspective to focus on static aspects also. The main reason for this is that an information system must store information about important entities, their properties and relations between them, which are all elements of the business context.

Action modelling and Action Diagrams are of interest in relation to the identification of important objects and the way that information about these objects should be structured. The reason for this is that information about essential actions, and their results must be structured and stored in the system. When performing static modelling, from an action-oriented perspective, both tangible things (e.g. cars and actors) and less tangible things (e.g. actions and messages) can be regarded as objects. The reason that messages are considered as objects is that messages are the results of communication acts, and presuppositions and triggers for subsequent acts. This implies a need to store information about, and to keep track of, important communication acts and messages produced.

When making static analyses of actions, it is of interest to describe both the actions and static dependencies between actor(s) and actions. For example, if we analyse the business offer, which is a communication act, and the static dependencies between the act and the actors involved, then we can end up with a UML Class Diagram (see, e.g. Booch et al., 1999) as shown in Figure 6-6.

![Figure 6-6: Relations between a communication act (a business offer) and the actors (a car dealer and a customer) involved in that act (Ågerfalk and Eriksson, 2002).](image-url)
From the Class Diagram we can see that the car dealer, the business offer and the customer are regarded as objects/entities. We can also see important static dependencies between these objects. The car dealer can be related to many business offers or none, and one business offer must be related to one car dealer. One business offer can also be related to one customer or none, and one customer can be related to many offers or none. These static dependencies are conditions or rules that must be derived from the business action context.

When we make a static analysis of a message, both the type of message (the illocutionary component) and the propositional content are of interest. In the diagram above, it is important to use the illocutionary component (the verb *offer*) to describe the type of communication act performed. It is also important to analyse the propositional content because it is used for referring to important objects. It is important to notice that the propositional content of the business offer refers to something that is called a purchase object, which refers to a car that the car dealer wants to sell. In this action context, not only can the car company sell cars that physically exist at the time when the car is offered but also the car company can offer cars that will be built after the customer has purchased the car (i.e. on customer order). The pragmatic meaning of an offer is not that it is true that the car referred to exists, which would be the case if the illocutionary component were used for stating a fact, for example, in the case of a *report*. The pragmatic meaning of the offer should be understood as an undertaking of an obligation to sell the purchase object referred to, whether it exists or not at the time the offer is issued. This condition has consequences for the conceptual model.

First, we cannot always use the licence number or the vehicle’s serial number as the key to refer to the car object, because these are identifiers used for cars that physically exist. Second, the attribute types and their values in the business offer may not refer to an existing car but can refer to a car that is to be built. Third, even if an existing car is offered, it can be the case that the car offered has different attributes from the existing car. This would, for example, be the case if the car dealer added attributes to the car in the offer, such as extra tyres and a stereo that were to be installed before the car was delivered to the customer.

Altogether, this implies that the existing car object and the purchase object offered are not really the same object; this is also obvious because we must allow for existing cars that have not yet been offered. Ultimately, this exemplifies the need for analysing the illocutionary component and propositional content together, and the need for the concept of *possible world* as discussed above. Figure 6-7 depicts this discussion.

![Figure 6-7: Relations between Business Offer, Purchase Object and Existing Car (Ågerfalk and Eriksson, 2002).](image)

From the Action Diagram in Figure 6-5 we can also see that different actions are related to each other. The business context is a network of actions where both material and communication are related to each other. For example, the offer is a presupposition for the purchase action, and the result of the purchase action is the purchase order. This implies that we must also describe interdependencies between different actions (see, e.g. the Class Diagram in Figure 6-8).
From the Class Diagram in Figure 6-8, we can see that a business offer can be related to many purchase orders or none, and that one purchase order must be related to one and only one business offer.

**Figure 6-8:** Relationship between different communicative actions (Ågerfalk and Eriksson, 2002).

In line with this discussion, the ‘complete’ resulting static conceptual model would consist of the classes Business Offer, Purchase Order, Purchase Contract (which would be related to the class Payment) and Car Delivery, all signifying important communication acts (see Figure 6-9).

**Figure 6-9:** The ‘complete’ resulting static conceptual model.

The conclusion to be made from this discussion is that the class Business Offer is an object type in its own right, (as is the Purchase Order, the Purchase Contract and the Payment). This is important to emphasize, as it may be tempting to view the Business Offer merely as a weak entity (e.g. as an association class), or as a state attribute of an order class, which can cause problems during database and system design and usage.
such as missing information and unintended deletion (see, e.g. Balaban and Shoval, 1999). This was also experienced in the case study where the sales support system was evaluated (Eriksson, 1998). In that particular system, the business offer was not represented as a regular entity relation in the database (i.e. it was not implemented as a separate relational table uniquely identifying offers). Instead, the offer was considered as a state attribute of the order class, and information about the order was updated by changing its status from ‘offer’ to ‘order confirmed’. This implied that when the offer was confirmed, and the business contract was established, the information about the business offer was no longer kept in the database. A problem caused by this solution was that the car dealers and the customers could not compare the offer with the purchase contract of the same car deal in situations when it was important to trace and compare these two generic business communication acts (cf. Goldkuhl, 1998). It also became difficult to make business analyses based on the information stored in the database. For example, it was impossible to see how many business offers had actually resulted in purchase contracts. This was believed to be essential information for analysing the effectiveness of the business. From this experience, we can learn the importance of considering important communication acts as object types in their own right, and that the change in state of the business context to Order Confirmed should not be modelled as change of a state attribute associated with an order class. The lesson learned is that it is essential to examine critically all important business communication acts to decide whether or not they should constitute regular entities about which the system must keep information, in order to construct an appropriate action memory (see Chapter 4). It is this author’s firm belief that generally they should do so, to promote conceptual models that are capable of handling changing business requirements.

Arguably, it would be possible to arrive at a static model that reflects these concerns without explicitly applying the proposed action-oriented perspective. However, the proposed approach aims to make this occur deliberately rather than by chance.

6.6 Pragmatization of Conceptual Modelling – Conclusions

The argument of this chapter is that (a) on the one hand that traditional conceptual modelling has focused too much on the semantic aspects of language and too little on the pragmatic aspects (see Section 6.2) and (b) on the other hand that speech act based modelling techniques have focused too much on the pragmatics of language and too little on the semantic aspects (see Section 6.4).

To remedy these shortcomings, action-oriented conceptual modelling based on speech act theory has been proposed. The ontological standpoint proposed implies that information systems can be used by business actors to perform actions and to store information about performed as well as anticipated actions – an action memory. We act in the world and manage information about action in the world. We must conceptualize and model the communication per se, not just the material world (or our conception of it). The propositional content of a language act can describe things that are yet to be, and we thus must understand conceptually and model both the existing world and a possible world.

Choosing speech act theory as a foundation for systems modelling is not a new concept. However, contemporary speech act based approaches, the CFA schema and Action Workflow being the most prominent examples, seem to have missed an important key notion within the theory, a notion that is crucial for the successful adoption of the theory as a foundation for conceptual modelling. In an attempt to incorporate the
intentional action aspect into business and system modelling, they have actually swung the pendulum too far and neglected the coupling between the propositional content and the illocutionary component of speech – what is talked about and what speaking does.

In this chapter, a focus on both material actions and communicative acts and the business context within which these actions are performed has been proposed. Based on an understanding of the dynamic structure of the business context, it has been shown how a static conceptual model of the business can be arrived at. Action-oriented dynamic modelling where social action is analysed within a business context was proposed. This analysis is performed from within (i.e. from the actors' perspective), and attempts to answer the question of what acting does (communicatively and materially). With this understanding as a base, it is possible to do action-oriented static modelling where social actions constitute conceptual objects about which the system is required to keep information. This analysis is performed from the outside, and attempts to answer the question of how the action is related to other things about which we must keep information. This way, the real strength of speech act theory as a foundation for conceptual modelling can be established. As a result, a foundation is laid for designing understandable systems that provide relevant information to users, and from which users can trace responsibilities for information, actions and commitments made.
Chapter 7

Usability in Social Action


7.1 Introduction to Usability in Social Action

The primary goal of using an automatic teller machine (ATM) seems rather obvious: to get some money out of it. Maguire (2001), for example, describes the ATM use-situation as consisting of a bank customer as primary user, the ATM as such (a computerized information system), and, among others, the task of obtaining money. However, if we look more carefully at this use-situation, we can observe that there is another actor involved – the bank, on whose behalf the ATM acts as an agent. Building on the notion of organizational action, introduced in Section 4.4.3, the bank, as an institution, can be seen as a social actor performing communication action made manifest by financial transactions taking place at the ATM user interface. Therefore it seems fruitful to view the use-situation from a social action perspective (Weber, 1978; Habermas, 1984). When taking social action (i.e. intentional behaviour that is oriented to the behaviour of others) as starting-point for understanding this use-situation, we must consider not only the instrumental goal of fetching the money, but also social goals which are related to social norms (Stamper et al., 2000). If the only thing that mattered was to efficiently obtain money, we did not, for example, need to design a system prohibiting unauthorized people from getting money from it. This means that we cannot only analyse, for example, how efficiently a user can use the ATM, we must also discuss whether it is used in a socially acceptable way. This is important to recognize because ATMs are built for honest and decent people, and they have features to protect them from misuse by dishonest people. Of course, since the tacit norms governing social action are a naturally institutionalized part of our life it is obvious for everyone that an ATM has to be protected from unauthorized use. However, it ought to be important to explicate such tacit norms and social goals related to them. This is important since norms and social goals affect how the whole use-situation, including the ATM, is designed, and as a consequence they are also important for understanding the usability of the ATM.

This chapter explores how the social action perspective of information systems actability can be used as an integrated complement to contemporary notions of usability (e.g. ISO 9241-11, 1998; Maguire, 2001). This is done by addressing how social goals are related to effectiveness, efficiency and satisfaction, three central criteria for the contemporary understanding of usability (ISO 9241-11, 1998; van Welie et al., 1999; Maguire, 2001) and quality in use (Bevan, 1999; 2001). Frøkjær et al. (2000) direct attention towards the correlation between these three criteria, and find it to be
rather weak. Based on this finding they conclude that ‘there is no substitute for including all three aspects in usability evaluations’ (Frøkjær et al., 2000, p. 351). Frøkjær et al. (2000) further argue that when using a narrower selection of usability measures, evaluators run an obvious risk of ignoring important aspects of usability. Inspired by that conclusion, this chapter directs attention towards the more fundamental question of the interpretation of these criteria altogether. Even though they are necessary, is the common interpretation of effectiveness, efficiency and satisfaction, as reflected in the contemporary usability literature (see below) really sufficient when viewing IT as a tool for business action and communication?

The discussion is exemplified and concretized through an analysis of the commonly recognized and understood case of the ATM use-situation. The aim is not to criticize the contemporary design of ATMs but to highlight the tacit norms implemented in them, and social goals related to them. Norms are important for all IT-systems (Stamper, 1996; Stamper et al., 2000) and the ATM has been chosen as an example since it is a commonly used system that most readers are likely to be familiar with.

The chapter is structured as follows. First, the contemporary understanding of usability following ISO 9241-11 (1998) is elaborated and argued to be too instrumentally oriented. Second, an alternative socio-instrumental view on IS use is elaborated. Finally, the ATM example is used in a critical examination of instrumental and social aspects of action as a basis for interpreting the usability criteria: effectiveness, efficiency and satisfaction.

7.2 The Contemporary Instrumental View on Usability

Arguably, one of the most important qualities related to the use of information systems is the usability achieved in actual use-situations (Bevan, 1995a; Maguire, 2001). One of the most widely adopted and cited definitions of usability is that by The International Organization for Standardization (ISO 9241-11), which identifies usability with the ability to use a product for its intended purposes: ‘the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use’ (ISO 9241-11, 1998). As reflected in the definition, three central criteria for usability are the effectiveness, efficiency and satisfaction with which users can achieve specified goals.

The first criterion, effectiveness, suggests that specified goals are to be achieved with accuracy and completeness (ISO 9241-11, 1998). As discussed in Chapter 1, effectiveness can be understood as the extent to which a system does what it is supposed to do (Preece et al., 2002, p. 14), which relates to the notion of ‘utility’ (Grudin, 1992).

The second criterion, efficiency, suggests that the expenditure of resources when achieving the specified goals should be minimized (ISO 9241-11, 1998). Put another way, measures of efficiency relate the level of effectiveness achieved to the expenditure of resources (Bevan, 1995a). According to Bevan (1995a), resources may be ‘mental or physical effort, which can be used to give measures of human efficiency, or time, which can be used to give a measure of temporal efficiency, or financial cost, which can be used to give a measure of economic efficiency’.

The satisfaction criterion suggests that users should have positive attitudes towards the use of the system, and feel comfortable with using it (ISO 9241-11, 1998). In this sense, satisfaction relates to concepts such as ease of use, user satisfaction, and usefulness (Davis, 1989; Mathieson and Keil, 1998).
Finally, the ‘specified context of use’ includes users, tasks, equipment, and the physical environment. ‘Task’ is here thought of as the activities required to achieve a goal (ISO 9241-11, 1998). Maguire (2001, p. 460) stresses the importance of the social context noting that ‘At a higher level, the attitudes of the organization and its employees towards the introduction of an IT system, and the way work is monitored, can affect whether a system is accepted and used to carry out the work. At a lower level the structure of the organization, the way people work (individually and in groups), the availability of assistance and the frequency of interruptions, are also likely to affect the usability of a product.’

Altogether, this view of user behaviour at the user interface of an IS can be compared with the traditional so-called teleological action model. In this model the actor uses various means (instruments) to achieve his or her goals, that is to accomplish desired effects. ‘When we describe behaviour as teleological action, we suppose that the agent reckons with an objective world in which he can know something and in which he can purposively intervene.’ (Habermas, 1984, p. 117) Actions are governed by action plans that actors choose based on their interpretation of the action situation and the goals to reach with the actions (Norman, 1988). According to Habermas (1978), such actions are founded in means/ends rationality.

Within the teleological action model, action is often interpreted as an instrumental act. That is, the focus is on the means/ends rational behaviour of an actor and the means he uses to achieve subjective goals. Of course, actions performed in interaction with information systems can be viewed as instrumental acts. To illustrate such an act we can return to the use of the ATM where the instrumental goal to be achieved is to obtain money. To achieve this goal, the action plan is to manoeuvre the ATM correctly (to push the right buttons in the correct order), the instruments used are the buttons on the ATM, the ATM card, the fingers to push with, et cetera, and the desired effect is that bills will eventually have been transferred from the machine to the user’s wallet.

From certain points of view, means/ends rationality is appropriate in order to understand human action (Weber, 1978). This is, for example, the case when an actor follows technical rules ‘mechanically’ to operate an IS to achieve an instrumental goal in the way described above. In other cases, it is too restrictive a perspective. The reason is that in most use-situations other actors and social values, norms and consequences must be considered, and even if not considered they are still present. Specifically, it is important from a systems development perspective to make sure that the means/ends rationality of a user conforms to the overall social context in which the user acts, even though potentially unaware of or uninterested in this larger context. This is where social action comes into play.

### 7.3 Social Action through Information Systems

Instrumental actions are often performed in a social action context, that is ‘determined by expectations as to the behaviour of objects in the environment and of other human beings’ (Weber, 1978, p. 24), which means that we have to consider them as social actions. Being oriented to the behaviour of others (people and/or organizations) requires an understanding of social norms (Stamper et al., 2000). Social norms are rules that govern social action and these rules are oriented to social goals and values (Weber, 1978). This implies that norms are social rules that are based on another type of goals and rationality compared to technical rules. Norms are social rules based on values and human behaviour and norms are not always tangible and ‘one cannot always put one’
hands conveniently on a norm. A norm is more like a field of force that makes the members of the community tend to behave or think in a certain way’ (Stamper et al., 2000, p. 15). Social norms are (like technical rules) a basis for achieving instrumental goals but the use of norms must be considered in a social action context. Furthermore, they are a basis for evaluating to what extent actions are ‘good’ or ‘bad’, that is for evaluating the quality of social actions.

One particular form of social action is action performed by use of language, referred to as communication action in this dissertation. In the theories of speech acts (Austin, 1962; Searle, 1969) and communicative action (Habermas, 1984), language is considered as an instrument for human communication and social action within a social action context. People perform communication actions to obtain instrumental goals. Nonetheless, perhaps one of the most important insights provided by speech act theory is that the use of language, and success of using it, is based on following a number of general rules (conventions), and that a communication action must be understood and evaluated within a social context. In Chapter 6, based on (Auramäki et al., 1988), this context was described as a combination of speaker, hearer, time, place and possible world. The possible world includes shared norms, values and beliefs and the existence of certain social and material (brute) facts. Note that this notion of social context is more fundamental to IS use than the notion of context suggested by, for example, Maguire (2001). The social context is not just a complicating factor that must be considered. The social context is what makes social actions at the user interface meaningful in the first place and, as such, is not just ‘likely to affect the usability’ (Maguire, 2001, p. 460), but a basis for understanding usability altogether.

Searle (1969, p. 69) claims that the speaker’s communicative intent with a speech act (a communication action) is to make the listener understand what he is trying to do by this act. Building on that notion, Habermas (1984) claims that the aim of communication is, in general, to create mutual understanding. According to Habermas (1979) the aim of reaching understanding implies bringing about an agreement constituted by reciprocal comprehension, shared knowledge, mutual trust and accord with one other. This implies that a speaker who performs a communication action, and who is oriented towards mutual understanding, must be able to raise four corresponding validity claims concerning the comprehensibility, truth, sincerity and rightness of the act (see Chapter 4). It also means that the listener must be able to evaluate, control and criticize the communication action based on these validity claims.

Searle (1969) has defined five pragmatic language functions which show five typical ways of using language, and goals (illocutionary points) which are related to these functions. The aim of a request, for example, is that it should count as an attempt to get the listener to perform a subsequent action (1969, p. 69). In a banking context, we can imagine that the customer walks up to the counter inside the bank and says ‘I would like to withdraw €50 from my account, please’. In this case, the customer performs a communication action of making a request. The customer must in this case follow the general rules valid for requests and the success of the speech act depends on how the customer performs the request and how the clerk interprets it, all within the actual social context. In order to succeed with his social act the customer must make the clerk understand how much money he wants to withdraw, and, most importantly, that he is authorized to make the request. The clerk must, in order to interpret this request, relate it to the actual social context. This implies that the request must be related to established social norms and procedures within the bank, and within this specific customer-bank relationship. There is, for example, probably a standard procedure to check cus-
tomers’ identity, which must be followed. When people behave like this, that is are oriented towards mutual understanding and conform to socially shared norms, we say that they are basing their actions on communicative rationality and perform communicative action (Habermas, 1984).

Communicative action is social action based on mutual understanding and thus conforms to mutually accepted social norms. Such action is not primarily oriented towards personal gains, even though mutual understanding and personal gains may very well coincide (Habermas, 1979). The crucial point is that the use of the ATM can be viewed as communicative action as well. Inside the bank, as well as at the ATM, the participating actors must live up to certain social goals which are based on social norms (such as making the customer understand that he must provide evidence of his identity) in addition to achieving the instrumental goals (such as withdrawing money). As discussed in Chapter 3, according to Weber (1978), rationality can be understood as a combination of means in relation to ends, ends in relation to values, and ethical principles in relation to action. This means that rational social action is always possible to relate to the means (instruments) used to achieve goals, and to values and ethical principles to which the action conforms. The first aspect can be referred to as an instrumental aspect of the action, and the latter as a social aspect. Together the two aspects constitute an important part of a socio-instrumental orientation towards acting, and are also important parts of the multi-functionality of communication acts discussed in Section 5.3.2.

Each of the two aspects, instrumental and social, is related to its own set of goals. The instrumental aspect is related to instrumental goals, which may be expressed in terms of achieving a given end. The social aspect is related to social goals, which may be expressed in terms of creating understanding and mutual agreement. It is important to see that social action includes both instrumental and social goals. This implies that we need to take social and not only instrumental goals into consideration when considering the usability of information systems – social goals are crucial for understanding the usability of the ATM.

### 7.4 Usability in Social Action – A Critical Examination

#### 7.4.1 The ATM example

Let us now return to the ATM example and examine how usability can be understood from a social action point of view. To that end, five typical actions performed in interaction with the system will be analysed. Certainly, the description constitutes an oversimplification, but is sufficiently detailed for the aim of this chapter. The ATM use-situation is visualized in Figure 7-1 by use of the Action Diagram notation introduced in Chapter 5 and further elaborated in Chapter 9. The Action Diagram utilizes an additional feature, rounded rectangles, to illustrate the main goals of each action, both social (dashed border) and instrumental (solid border). It is important to recognize that all actions embody both an instrumental and a social aspect, which implies that:

34 Of course it is also possible to take norms and values into account to deliberately violate them for personal gains. Such strategic action (Habermas, 1984) may be important in order to understand systems development as an activity (Hirschheim et al., 1996). Even more important to see is that in most moral-practical senses, supporting such actions should not be a goal of systems design (cf. Ljungberg and Holm, 1996).
the communication actions (Actions 1, 2, 3 and 5), can be related to both social and instrumental goals, but since they are socially oriented this implies that social goals should be emphasized when the quality of the acts are considered.

The quality of the material action (Action 4) should primarily be considered based on the instrumental goal of handing out money even though the action embodies also a subordinated social aspect.

Figure 7-1: Action Diagram showing the ATM use-context (Ågerfalk and Eriksson, 2003).

Action 1 – Input Card Request: The first action is a communication act in which the bank requests that the customer put the bankcard into the machine and enter a pin code. The social goals of the act are to make the customer understand that he is being offered to use the ATM service, and that he must provide evidence that he is allowed to (which is based on a previous agreement between the customer and the bank). This is thus a multifunctional communication act (see Section 5.3.2) involving both an offer and a request. The instrumental goal of the bank is to communicate these messages to the customer through the ATM user interface. If the customer understands this he can proceed to action 2.

Action 2 – Enter PIN: The second action is a communication action in which the customer puts the bankcard into the machine and enters the associated PIN code. The
Social goal of this action is to make the bank understand that the customer is authorized to use the ATM. This understanding rests on two conditions: (a) the customer has access to an authorized bankcard to identify himself to the bank; (b) the customer knows the PIN code associated with the bankcard. If these conditions are met, that is, if the bankcard is valid and the provided PIN code matches the bankcard, the bank gives the customer access to the ATM’s functionality. The instrumental goal of the customer is to manage to insert the card correctly and enter the corresponding PIN code.

Action 3 – Request Money: The third action is a communication action in which the customer requests that the bank hand out a specified amount of money. The social goal is to make the bank understand that the customer wants the bank to hand out the money. This understanding rests on two conditions: (a) the customer has the right (i.e. is authorized) to make the request (this condition has already been tested during Action 2), and (b) the account balance is sufficient to cover the requested amount of money. If the bank (through the ATM-system) accepts that these conditions are fulfilled then the bank can perform Action 4, otherwise Action 5 is performed. The instrumental goal of the customer is to manoeuvre the ATM appropriately so that the request eventually reaches the bank.

Action 4 – Hand Out Money: The fourth action is a material action in which the bank hands out the money to the customer. The instrumental goal of the act is to have the specified amount of money handed out. The money transfer depends on the condition that the ATM contains the required amount of bills. If the customer gets the money and verifies that the money received corresponds to the amount requested, then the customer is probably satisfied. Consequently, the social goal of the act is that the user understands and accepts the money as a response to his request. If the required amount of bills is not available, the customer is informed (by a further act) that there is not enough money in the ATM to hand out the money. The social goal of this further action is to make the customer understand that the ATM is out of bills.

Action 5 – Abort Transaction: The fifth action is a communication action in which the bank communicates to the customer that the bank cannot hand out the requested amount of money. The communicative goal of the action is to make the user understand that the bank cannot hand out the money. This understanding rests on the condition that there is an agreement between the bank and customer committing them not to create a negative account balance. If the user understands this condition he understands why the bank is not allowed to hand out the money. The instrumental goal of the bank is to have the information communicated to the customer through the ATM user interface.

As we can see, Actions 1, 2, 3 and 5 are oriented towards the overall social goal of creating a mutual understanding, while Action 4 is oriented towards the customer’s desired output in terms of getting the money out of the machine. What is important to emphasize from a social action point of view is the importance of the social goals and their role in ensuring that the actions are performed not only in an efficient but also in a socially acceptable way. For example, Actions 1 and 2 are mainly performed in order to secure that the transaction is performed based on social norms governing the interaction. What is in focus here is whether the customer is allowed to use the ATM, not the instrumental goal of fetching the money. The social goal of Action 3 is related to whether the customer is allowed to withdraw the requested amount of money, and this action rests on a prior mutual agreement between the customer and the bank that the customer is not allowed to create a negative balance in his account. These social goals are essential for the design of the ATM and how the customer experiences the use-situation. If the customer, for example, tries to make a request that would have created
a negative balance on his account, it is important that he understands that he is not allowed to withdraw this amount. It would probably be hard for the customer to trust a bank with an ATM that did not check if the request would create a negative account balance, or even worse, did not check if the user were authorized to use the ATM.

Analysing the use-situation along these lines is crucial from a social action-perspective since social action is concerned not only with how to perform actions in an efficient way. It is also concerned with the goodness and the moral-practical rationality of the actions, and the trust of social actors (individuals as well as organizations). It also implies that in order to understand the use-situation, both designers and the actors that use the system (i.e. the bank and the customer in this case) must have a thorough understanding of the social context in which the system is being used. It is not only a matter of reaching the instrumental goal of getting the money out of the machine; it is also a matter of whether this is done in a socially acceptable way. It is therefore insufficient to analyse the ATM case in terms of satisfaction, efficiency and effectiveness with an instrumental orientation alone, these criteria must be interpreted based on a socio-instrumental orientation including also the social aspect of action.

### 7.4.2 Interpretations of Effectiveness, Efficiency and Satisfaction

With an instrumental orientation, the human use of signs is abstracted in a way that information is reduced to an instrument for ‘triggering and maintaining goal-seeking behaviour to achieve pre-specified outcomes in a controlled environment’ (Hirschheim et al., 1996, p. 19). From this perspective it is highly relevant to speak of effectiveness and efficiency only in terms of desired output and relative expenditure of resources, and satisfaction as comfort and positive attitudes towards a system. If we choose to broaden the view and take into account also social aspects, we can interpret the three usability criteria: effectiveness, efficiency and satisfaction, in a more elaborated way (see Table 7-1).

#### Table 7-1: Interpretation of effectiveness, efficiency and satisfaction with a socio-instrumental orientation.

<table>
<thead>
<tr>
<th>Usability criteria</th>
<th>Socio-instrumental orientation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Instrumental aspect</td>
</tr>
<tr>
<td>Effectiveness</td>
<td>Desired output</td>
</tr>
<tr>
<td>Efficiency</td>
<td>Relative expenditure of resources</td>
</tr>
<tr>
<td>Satisfaction</td>
<td>Comfort and positive attitudes</td>
</tr>
</tbody>
</table>

The first criterion, effectiveness, suggests that specified goals are to be achieved with accuracy and completeness (ISO 9241-11, 1998). Let us turn to the MUSiC (Metrics for Usability Standards in Computing) Performance Measurement Method (Bevan, 1995a) to get an example of the operationalization of the criterion. According to this method measures of effectiveness relate goals of using an IS to the accuracy and completeness with which these goals can be achieved. Completeness, the amount of a task completed by a user, is related to Quantity and Quality and is a measure of the degree to which the output achieves the task goals. In the ATM example, completeness would correspond to whether the user managed to withdraw money (Quantity) and to the match between the amount requested and that received (Quality). Both measures are
expressed as percentages and are used together to calculate the effectiveness of a task as: Task Effectiveness = 1/100 (Quantity \times Quality) \% (Bevan, 1995a)

With a socio-instrumental orientation, information systems should be designed as to facilitate mutual understanding. The main point of a socially oriented goal such as mutual understanding is that it is the validity of the goal that should be evaluated not the task effectiveness in terms of percentages. Therefore it is not feasible to evaluate mutual understanding with formulas such as the one presented above. Of course actions must also contribute to task effectiveness, but this is not enough (it is a necessary but not sufficient condition). In order to understand the usability of an IS we must consider the validity and meaning of the social goals and the social actions performed by use of the system, which is another matter. This implies another way of interpreting and evaluating effectiveness. As a consequence we must learn how to evaluate social actions based on their meaning and validity. In order to understand the meaning of social actions we have to know under what conditions (as derived from the social context) they are acceptable based on validity (Habermas, 1984, p. 115).

The second criterion, efficiency, suggests that specified goals are to be achieved with as little expenditure of resources as possible (ISO 9241-11, 1998). Following this definition, efficiency is calculated in the MUSiC Performance Measurement Method as (Bevan, 1995a): Temporal Efficiency = Effectiveness / Task Time.

Mutual understanding is hard to relate to the expenditure of resources and therefore it seems meaningless to speak of efficiency in relation to social goals. Of course, communication and the performance of action by use of language, altogether, can indeed be analysed in terms of efficiency. In this respect it is important to see that the performance of a communication act includes both an instrumental and a social aspect. That is, when performing a communication act at the interface, we also use the system (and our language) in an instrumental way, and this use can be related to the relative expenditure of resources.

The third criterion, satisfaction, is the subjective criterion used for describing and measuring the actors’ (users’) feelings and attitudes towards the system and the achieving of the goals that make a system effective. Habermas (1979) claims that an agreement based on mutual understanding must include mutual trust. In order to act with the aim of creating mutual understanding, the actors must rely on a social context which includes actors, norms and institutions. Trust is essentially based on faith in others’ word and the legitimacy of regulations, and underlines the importance of belief in other people (Salatín and Flores, 2001). When relating the criterion of satisfaction to a socio-instrumental orientation, satisfaction is associated with the inter-subjectivity of social action – the trust communicating actors have in each other as well as in the system and the actions performed through and by means of the IS (cf. Cardholm, 1999). This way, satisfaction is not only a property of a user’s interaction with a system but also of an actor’s participation in a social action context.

7.5 Usability in Social Action – Conclusions

This chapter has suggested taking social action as theoretical point of departure for understanding the usability of information systems. By use of an ATM example, it has been shown how the traditional criteria used to understand IS use in terms of effectiveness, efficiency and satisfaction can be reinterpreted from a social action perspective.
When information systems are used in a social action context they can be used to perform communication action. All communication actions have both an instrumental aspect and a social aspect and it is possible to extract complementary success criteria for such actions from different perspectives. In order to understand the usability of an IS, it is important to not only consider a ‘cognitive instrumental rationality, concerned with the evaluation of objective facts’ (Ehn and Löwgren, 1997, p. 308). That is, we can choose to view an action from an instrumental perspective. In this case, the success of the action can be judged based on achievement of desired output, relative expenditure of resources and subjective comfort and positive attitudes towards the IS. The same action can also be viewed from a social perspective. In that case, the successful performance of the action can be judged based on established mutual understanding and achieved inter-subjective trust.

A problem with the contemporary understanding of usability is that it is often expressed in terms of achieving goals, which, at least tacitly, seem to be restricted to goals restricted to the instrumental aspect of social action, as can be seen in, for example, the work of Bevan (1995a) and Maguire (2001). The argument of this chapter has been that in order to fully understand usability, we must consider both instrumental and social goals since their combination constitute a fundamental part of the social action context in which systems are used, and both instrumental and social goals affect the way systems and use-situations are designed and conceived.

Designing usable (and actable) IT-systems requires understanding the social action context in which systems are going to be used, which is more than understanding users’ instrumental goals and an objective context of use. Rather, it is to understand social norms and commitments and related social goals that govern social action. This implies that designers must not only tacitly have a feeling for social norms governing the use of the system but that such norms must be explicated and analysed during design. This is necessary in order to achieve usability in social action by intention rather than by chance. It would, for example, be quite possible to view the social goal of making the bank understand that the customer is authorized to use the ATM (see Action 2 in the ATM example above) only as a sub-goal to the instrumental goal of having the ATM handing out the money, that is, only something that has to be done in order to get the money. However, if these two goals are viewed as two separate goals belonging to different goal classes (social and instrumental), and a general rule is that all goals of both classes should be met, we do not run the risk of incorrectly viewing the instrumental goal as the ultimate, with the risk of ignoring the social goal. This is important since social goals and the norms related to them govern the interaction and are an important prerequisite for understanding the usage quality of IT-systems.
Part III

Actability Design
Chapter 8

Research on Actability Design

As described in Chapters 1 and 3, the research on Actability Design has centred on the development of method support for requirements engineering (RE) and information modelling. The concrete outcome of this work is a redesigned development method termed VIBA (Versatile Information and Business Analysis). This chapter describes how the redesign of this method has been conducted, which should be understood in the light of the discussions in Chapter 3 and Section 2.3.5.

8.1 Actability Design Background

There are many different methods and approaches to support systems development. Even though they all aim at the same thing – building ‘good’ information systems – they often differ in several ways. Sometimes the differences are subtle (for example different object-oriented methods that share the same concepts and notations) but sometimes the underlying values and perspectives are certainly discriminating, i.e. the methods’ inherent notions of what constitutes a ‘good information system’ and how systems should be built differ significantly (see Section 2.3).

VIBA was originally developed by Göran Goldkuhl, partly as a reaction against shortcomings identified within the ISAC method (Lundeberg et al., 1981) with which he had worked through several years. The method was first published in a Swedish book: Verksamhetsutveckla datasystem (Goldkuhl, 1993). The method was well received within academia and has been used extensively in informatics education at several Swedish universities ever since. One of the most interesting benefits of VIBA was its foundation in ‘human infology’: a theoretical perspective that grew out of the systemic infology of Langefors (Langefors, 1973), emphasizing the linguistic and social character of information systems (Goldkuhl, 1980; Goldkuhl and Lyytinen, 1982). Throughout the years, VIBA has also been used in industry to some extent.

The main problem targeted by VIBA was that information systems and the businesses they should operate in seem to be disharmonious. Users feel that information systems are hard to understand and regard them as belonging to the IT department instead of their own workplace, i.e. they feel alienation and helplessness; this is, according to Computer Sweden (1999), still a pervasive problem. Customer value is often neglected and information systems are often used as an excuse for poor customer service instead of being regarded as service enablers. To solve these problems VIBA promotes a business-oriented approach (‘verksamhetsinriktad’ in Swedish). That is, the business and the business’ actors are in focus when developing systems. Systems development is considered a special kind of business development and the aim is to build information systems that:

- are easy to use,
- are integrated and congruent with the business,
- are understandable and predictable,
- have visible actors, and
- enable and support good working conditions.
Over the years, the method has been criticized and challenged in several ways. The sources of criticism represent both theoretical work, as seen in several evaluations made by students during systems development classes and examination projects (e.g. Vallgren, 1992; Överström, 1994) and more practical assessments, as in, for example, the fact that different instructors have modified the method to better fit their particular needs (e.g. Nilsson et al., 1997).

One criticism concerns the complex handling of user interaction modelling and the lack of (or at least unclear) support for experimental and iterative development. It has also been challenged as not making use of the benefits of object orientation, which has attracted more and more interest (and success) during the last couple of years. Another criticism, which can be directed towards many systems development methods, is unclear statements on the applicability of the method, i.e. the method context (see below for discussion of method context). With this in mind, in the early autumn of 1997 we started working on ways to re-design VIBA. We wanted to make the method even more explicit about its foundations in a further developed action theory of information systems (which evolved into the concept of IS actability), and to take advantage of recent developments and (successful) trends within the areas of information systems, software engineering, and human-computer interaction.

The reason for taking VIBA as the point of departure for this work was VIBA’s previously discussed foundation in human infology, which matched our intentions of emphasizing the action character of information and information systems, i.e. that to conceive information systems as information action systems is a key factor in the success of systems development. Of course, some other method based on similar theoretical foundations, such as, for example, the Commodious method (Holm, 1996) or DEMO (Dietz, 1994), would have sufficed. The reasons for favouring VIBA, in this respect, were simply the availability of the method and the method’s rationale; the creator of the original VIBA was actually part of the research project, which, of course, influenced the choice.

**8.2 Actability Design Assumptions and Design Goals**

The re-designing of the method has been based on two fundamental assumptions. The first assumption is that conceiving information systems as information action systems is a key factor in the success of systems development. By viewing information systems and information systems development as mainly social phenomena, in the sense of communicative social action as summarized by the IS actability concept, we begin to understand the nature and problems of systems development and are, hopefully, able to cope with them appropriately.

The second assumption is that information systems are subject to change at an ever-increasing speed. One implication of this is that specified requirements are valid for shorter and shorter periods in order to keep information systems up-to-date with rapidly changing business environments.

This work has also been deliberately influenced by five recent and, arguably, important knowledge domains of information systems. These are:

- business process orientation,
- object modelling (object orientation),
- usability,
- rapid (application) development (including rapid prototyping) and
- software requirements management.
The following section presents these knowledge domains and their suitability for a re-designed VIBA is argued. The presentations follow a two-phase structure. First, a description of what the domain is and why it is important to consider it in systems development (our intention in using it) is given. Second, how the domain affects the re-design of VIBA (its intended use) is addressed.

8.2.1 Influential Knowledge Domains

8.2.1.1 Business Process Orientation

Business process orientation (BPO) means that businesses are viewed as consisting of value-adding processes that cut through traditional functions. It is the customers that judge the amount of added value and BPO thereby implies a strong customer orientation. Davenport (1993) explains the difference between traditional hierarchical organization and process organization by asserting that: ‘Whereas an organization’s hierarchical structure is typically a slice-in-time view of responsibilities and reporting relationships, its process structure is a dynamic view of how organizations deliver value’. However, as pointed out by Jacobson et al. (1994), the concept of process is nothing new. Within organizations there have always been processes. Applying BPO simply implies that the business’ processes are emphasized and taken as a starting point for business modelling rather than the traditional hierarchical division into functions. Nevertheless, a functional division might very well exist in parallel with a process organization. A reason to favour BPO is according to Jacobson et al. (1994, p. 3) that ‘Processes arise more naturally than hierarchies because they come into being when people realize that they must cooperate to achieve the result promised to the customer’. Hence, taking business processes as a starting point for business design would yield (1) a better match between formal and informal structures, and (2) a smoother implementation of the new design. Another argument for BPO is that it makes it easier to estimate and measure, for example, cost, time, quality, and customer satisfaction (Jacobson et al., 1994).

The applicability of a method is closely related to the problem domain(s) it addresses. This applicability might also depend on desirable types of solutions to problems in a certain domain. For example, a function-oriented method such as Structured Analysis (Yourdon, 1989) is probably better suited for adoption to a function-oriented organization than a workflow approach, even if they share the same problem domain (business information systems development). These two aspects of methods can be referred to as the ‘method context’. The method context was previously not well articulated in VIBA. By re-designing it we wanted to be more explicit. The method context of VIBA is business action supporting information systems within process-oriented organizations.

8.2.1.2 Rapid Development

A method is always used together with a framework. A framework is a high-level structure that dictates what to be done without the details of, and specific techniques for, how to do it (see Section 2.3.2.4). Martin (1991) presented an alternative to traditional development models and, in doing so, he coined the expression ‘Rapid Application Development’ or RAD for short. According to Martin (1991) RAD:

35 We will in this dissertation use the terms ‘framework’, ‘development model’, ‘process model’ and ‘lifecycle model’ interchangeably. See Section 2.3.2 for a discussion of these concepts.
Information systems actability

- avoids specifications becoming obsolete before cutover;
- needs user involvement in design workshops;
- needs prototyping, which helps meet user needs;
- needs user involvement in the construction phase;
- needs CASE tool support;
- needs code generators that produce bug-free code; and
- tests the evolving design concurrently with construction.

Martin (1991, p. 5) further argues that RAD will lead to higher quality, and that ‘high quality, lower cost and rapid development, thus, go hand-in-hand if an appropriate development methodology is used’. In this context, Martin defines quality as: ‘meeting the true business (or user) requirements as effectively as possible at the time the system comes into operation’.

Rapid development means ‘developing information systems fast’, following McConnel (1996), and different development models can be used to support such development. Rapid development is often associated with incremental development models, such as Boehm’s (1988) spiral model, and tools for rapid application development, such as Borland Delphi and Microsoft Visual Basic, used with a prototyping approach (e.g. Smith, 1991). But, as McConnel (1996) points out, which development model to choose depends on the development situation and available tools that support rapid development. Consequently, a crucial part of any development project is to choose a model that suits the demands of that particular development situation. For instance, a pure waterfall model might well do if requirements are well understood at the beginning of development and reliability is of high importance. The particular development model that McConnel claims will fit most rapid development projects is the spiral model (Boehm, 1988), which, in some sense, is the direct opposite of a pure waterfall. With a spiral model projects start small and expand in scope in increments. The scope is not expanded until the risks for the next increment are reduced to an acceptable level.

One recent approach to rapid development is the Dynamic Systems Development Method (DSDM) (Stapleton, 1997). DSDM criticizes traditional approaches for fixing requirements early while allowing time and resources to vary during development. In DSDM the opposite is true. While letting functionality vary, time and resources are considered as being as fixed as possible.

Thus, rapid development means that the best products possible, given clear limits on time and resources, are what to strive for, and hence not ‘perfect’ products in a traditional sense (i.e. with all requirements fulfilled).

No matter what approach to rapid development one chooses: in order to keep information systems up to date in an ever-changing business environment, information systems must be developed as fast as possible – without reduced product quality, including documentation.

8.2.1.3 Object Modelling

A concept that has gained more and more attention during the last ten to fifteen years is that of the ‘object’. In entity-relationship (E/R) modelling (Sundgren, 1973; Chen, 1976) an object (entity) represents a concrete or abstract phenomenon in the problem domain that the system is supposed to keep information about. The object consists of relevant information about the real world phenomenon, which can be operated on by functions and procedures as the result of queries and updates. In object orientation (Coad and Yourdon, 1991; Booch, 1994) the functionality is considered encapsulated
in the objects and it is usually said that objects ‘inhabit behaviour’. Industrial experience shows that object orientation is a preferable way to organize and design software systems. Lately, there have been proposals as to how to use object orientation also during business modelling (e.g. Graham, 1998; Kruchten, 1999).

Due to object orientation’s inherent modularization of software systems, it seems to lend itself well to the construction of systems that are subject to frequent changes, following the principles of Parnas (1972), in that it helps structuring requirements management. As pointed out above, one key factor of rapid development is the use of automated tools and reusable parts. Contemporary tools for rapid development and graphical user interface programming (RAD-tools) are usually based on object-oriented technology. Hence, to implement rapid development it is in practice necessary, in one way or another, to make use of object-oriented technology, at least during technical implementation. Nonetheless, the object-oriented way of encapsulating behaviour and knowledge into congruent wholes (objects and classes) leads to problems when using today’s RAD-tools due to their strong bias toward the relational model of data (Ågerfalk, 1999a). Therefore we have chosen not to enforce object orientation during analysis but to provide a smooth transition path from requirements analysis to object-oriented analysis and design.

8.2.1.4 Usability

Advocates of usability emphasize that computerized systems should be usable when performing interactive tasks (see Section 1.2.1 and Chapter 7). At the same time, traditional methods for IS development suffer from limitations in their treatment of cognitive and human factors, as well as in their recommendations for the analysis of different interaction styles and how to cope with differences in the experiences of different users (Lif, 1998). These topics are typically not stressed in the information systems development field and have so far been completely left out of the approaches based on the language/action perspective. Hence, it seems fruitful and important to combine an action-oriented perspective with that promoted by usability, as was shown in Chapter 7.

When designing IS interaction, usability is important to consider. However, the notion of usability often seems to be too narrow, partly because it is often perceived as dealing only with how to design user interfaces (Holmlid, 2002) and mainly because it often brings with it the overly instrumental view of human action discussed in Chapter 7. When designing communication through an IS, the question of how to interact is, of course, important. Equally important, however, is what to communicate and why. Moreover, all three aspects must be considered in a context where the communication is taking place – that is, a social context that is never static and fully predictable. Thus, usability promotes an important perspective but needs to be extended and reinterpreted in order to put more emphasis on the business context in which the interaction is taking place. Information systems should be usable not only in the context of interaction but also in the context of business action.

8.2.1.5 Software Requirements Management

Software requirements management has emerged as an approach to handling changes in requirements and to helping predict changes to software (and change costs) in response to business changes. The basic approach is to maximize traceability of requirements (Lam, 1998). This can be done in two different (non-orthogonal) dimensions. The first dimension concerns traceability from business models to detailed system design models. The other dimension cuts across models at the same level of abstrac-
tion. Requirements management is important in order to ease the burden of maintenance and further enhancement. Since information systems are assumed to be subject to frequent changes, some method support is needed for rapid enhancement as well as for rapid development. By taking this into account from the very start of systems specification, a better foundation for change is built.

8.2.2 High-Level Re-Design Goals

We have had three high-level re-design goals when working with the re-design of VIBA. The first goal was to make the theoretical foundations even more explicit and to take full advantage of their consequences. The theoretical foundation has been formulated as the concept of IS actability. The second goal was to design a method that produces documentation that is communicable between developers and other stakeholders. The third goal was to make use of as much existing knowledge and good practice as possible. The latter includes reuse of existing modelling formalisms and notations whenever possible; i.e. so that they are possible to integrate with the proposed actability view of information systems.

8.2.3 Requirements in Context

It is a well-known fact that information systems often fail to meet the requirements of their users. Hence, the specification and evaluation of requirements is a crucial part of information systems development. The management of requirements is a difficult task. It is hard to elicit the appropriate requirements and to formulate them in such a way that they are understandable to users and formal enough for programmers to build software.

There are many approaches to systems development and RE. There are, for example, functional methods such as Structured Analysis (Yourdon, 1989), information engineering methods (Martin, 1990) and object-oriented methods such as the Rational Unified Process (Kruchten, 1999).

All these approaches aim at modelling requirements. Advocates of prototyping (e.g. Smith, 1991; Budde, 1992) have challenged such modelling as being too abstract. It is hard for users to understand how an information system will function from abstract models (such as data models and data flow models). The use of simple prototypes of software is claimed to enhance the understanding of future systems among users. Through experience of the behaviour of the future systems, users will come to an understanding of what it might be to use those systems. Prototypes are often used as an aid for designing user interfaces. The underlying principles of the system are however harder to visualize in a prototype. Abstract requirement models are important to use for description, analysis and critique of principles and features that go beyond the user interface. Therefore, reconciliation is needed between modelling and prototyping (c.f. Mathiassen et al., 1995). In RE there is a need for, on one hand, abstraction and modelling and, on the other hand, visualizing and experiences.

---

One important lesson to learn from the prototyping versus modelling debate is the need to go beyond the static properties of a system. Many system models tend to model only structural and static aspects of systems. Dynamic and active aspects of systems need also to be treated and modelled during RE (see Chapter 6). This is also apparent if we look at the evolution of methods for RE. Classical methods for data modelling, such as the entity-relationship approach (Chen, 1976), had a focus on fairly static aspects. Object orientation can to some extent be interpreted as a critique of such a static emphasis. In object orientation there is an attempt to model both static and dynamic aspects of IS (e.g. Rumbaugh et al., 1991). The further development of object orientation seems to go into more dynamic aspects of information systems. The use case notion (Jacobson et al., 1992) and other scenario-based approaches (e.g. Carroll, 1995; 2000) constitute attempts to capture the intended behaviour of a system from its outside.

The historical evolution of RE shows an increasing interest in information systems as dynamic and active entities. Both rapid development (including prototyping) and object orientation are examples of this evolution. Unfortunately these approaches, although having many good features, lack a proper understanding of information systems as organizational phenomena. They fail to see the genuine social character of information systems. Information systems are important instruments for organizations to perform their actions. Information systems are systems for action.

RE is really a ‘never-ending story’. The story begins with some fuzzy ideas about how an IS might support the way business is done (or perhaps ought to be done). The process of eliciting and formalizing such ideas into a requirements document (software requirements specification) is generally referred to as requirements elicitation or requirements definition and specification (Sommerville, 1996). It is important to see that the RE process does not end with such a document. During the life of an IS, requirements are likely to change many times in order to keep up with an ever-changing business environment. Thus, requirements change management becomes an important part of the entire RE process (cf. Lam, 1998; Lam et al., 1998).

### 8.3 Actability Design Research Approach

An obvious, but naïve, research process could have been performed in three stages. Firstly, the concept of IS actability could have been constructed (articulated and externalized) based on social action theories and other influences (such as business process orientation). Secondly, actability could have been used as the theoretical foundation for the re-design of VIBA. Finally, empirical tests of VIBA could have been performed to verify both VIBA and actability. Following such a research process would have stopped us from letting empirical work influence the design of actability. Hence, in the spirit of the hermeneutic reflective research approach described in Chapter 3 we wanted to be open for changes in the design of both actability and VIBA as far as possible. The main message of hermeneutics is that knowledge development is cumulative and that the understanding of a certain phenomenon depends on the observer’s pre-knowledge at the time of observation. Thus, by re-evaluating the ‘current’ understanding with the new understanding as pre-knowledge, we gain further knowledge of the phenomenon studied.

Following the discussion in section 2.3.5, the re-design of VIBA has been performed in two primary focal areas. The first main focus was method construction, and internal and external theoretical grounding. The second main focus was empirical grounding, which also led to further method development and internal grounding.
As initial inputs to the work there was the existing VIBA, the previously described prerequisites in terms of assumptions, design-goals and influences, and the theoretical foundations of information systems as information action systems. We thus were in a position that lent itself to a mixed approach of deductive and inductive method construction – a reflective approach (see Chapter 3). We had the existing VIBA with some prescriptions that had proven useful and which we wanted to keep. This is an example of an inductive approach, even though we did not have to reconstruct the method since it was already documented. On the other hand, we had the prerequisites that had to be incorporated in the new VIBA deductively.

The empirical work was performed in five different cases wherein VIBA was applied (referred to as Applications 0–5, see Section 8.4 for details). The approximate positioning in time of these cases is shown in Figure 8-1.

![Figure 8-1: Approximate Time span of the different applications of VIBA (A0 – A4).](image)

The initial work was carried out through modelling and analysing an already existing system (Application 0). During that work we tried different suggestions and techniques and performed an integrated internal and external validation process.

The subsequent work was carried out as two development projects in collaboration with industry, referred to as Applications 1 and 2. During specification of a production planning system for mechanical engineering (Application 1), and specification and realization\(^{37}\) of a material administration system for a paper mill (Application 2), the new method proposal was tried in industrial settings and continuously refined to comply with real life demands. The two applications served somewhat different purposes. When Application 1 was initiated we had only some vague ideas about the re-designed VIBA. The purpose was hence to test initial ideas and continuously refine VIBA and actability. When Application 2 was initiated the method was quite stable (the frequency of changes based on experiences from Application 1 was approaching zero). The purpose was now to test, more thoroughly, the re-designed VIBA and to be more careful with documenting observations. Hence, whilst Application 1 was more of an explorative study, Application 2 was more of hypothesis testing. Nonetheless, both VIBA and actability continued to change during both applications.

At this point, VIBA had matured sufficiently (it had been tried in two applications and was documented in the author’s licentiate thesis (Ågerfalk, 1999b), which led to several suggestions and modifications) and it was documented in a compendium (Ågerfalk et al., 2000a) and used in an intermediate systems development course at Örebro University, where the author of this dissertation was employed as a lecturer.

---

\(^{37}\) In this context, ‘realization’ refers to the actual construction of software, i.e. programming.
The teaching of the method and its underlying theoretical base (actability) to students served as a test of the communicability of the method and to reveal possible inconsistencies. This teaching experience, during which the course was given five times, is referred to as Application 3.

A further test of the method was performed as a development project in collaboration with a Swedish municipal elderly care unit (see Cronholm and Goldkuhl, 2002). During this Application 4, this author did not participate actively, but performed a post development evaluation consisting of interviewing the participants in order to get their opinion about working with the method.

As discussed in Chapter 3, three different levels of interest embracing different units of analysis have been distinguished in this research: the levels of theory, practice and approach. At the practice level, the research is concerned with interacting users use of information systems to perform business actions within a specific type of social action context referred to as the business context. These three components constitute different units of analysis, which must be understood in relation to each other; for each unit the other two constitute its context (see Figure 8-2).

Figure 8-2: Units of analysis in the actability design research.

The specification of information requirements and visual appearance of information systems have been performed by use of an explicitly stated and evolving development method (VIBA). That method, per se, constitutes a further unit of analysis in the study. For the purposes of this dissertation, the interest is in the properties of the approach and to what extent it directs attention to, and helps to explain, relevant phenom-
ena within the practice studied. That is, how it helps the different development actors (see Figure 8-2) to create an understanding of the IS being developed at the level of practice. This includes the activities proposed by the method, the concepts used to model the system and its surrounding, and the forms of documentation used in the method. At the level of theory, the research is concerned with the concept of actability and the repercussions the development of knowledge of the other two levels has on it.

With respect to particular information systems, the research interest concerns specification of the systems’ future design. The interest in different individual interacting (future) users is related to their subjective understanding of the future system and its relation to the business context and the business actions performed within it. Specifically, interest is paid to future user’s understanding of the VIBA documentation and thus to the communicability of the different concepts and forms of documentation used. The business context is interesting from the point of view that it frames system use and the way the system and its use contribute to it. Such understanding is important to estimate the alignment between the system and the business it supports. Studying VIBA in use (as existing on the action level, see Section 2.3.4) is imperative to understand to what extent the method and its components aid in directing attention to and specifying the system. Such understanding is imperative since it is a natural part of the evolution of the method (see the discussion about data and ideas below). Finally, the primary interest is directed towards actability as a concept; to what extent actability provides a relevant understanding so that the system, within its business context, can be appropriately specified. Table 8-1 summarizes the interests in the different units of analysis.

Table 8-1: Interests and intended results in relation to units of analysis.

<table>
<thead>
<tr>
<th>Unit of analysis</th>
<th>Main interest</th>
<th>Intended results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Information system</td>
<td>Specification of the system’s features and appearance</td>
<td>System specification</td>
</tr>
<tr>
<td>Individual interacting users</td>
<td>Subjective understanding of developed system and its business context by use of VIBA as well as an understanding of the system’s intended users</td>
<td>Understanding of individual differences regarding the understanding and appreciation of the proposed VIBA documentation</td>
</tr>
<tr>
<td>Business actions</td>
<td>What actions are performed and why</td>
<td>Understanding of the future (intended) use of the system</td>
</tr>
<tr>
<td>Business context</td>
<td>The system’s support to the business context and the actor’s roles within it</td>
<td>Basis for understanding the role of the system and estimation of the alignment between the system and the business it supports</td>
</tr>
<tr>
<td>VIBA</td>
<td>To what extent the method and its components aid in directing attention to and specify relevant aspects</td>
<td>Experience of working with the method and method evolution</td>
</tr>
<tr>
<td>Requirements engineers, programmers, and system owner</td>
<td>Comprehension of VIBA and its support for understanding the practice level and specifying the system</td>
<td>Understanding of the method’s support and incentives for method refinements</td>
</tr>
<tr>
<td>Actability</td>
<td>To what extent actability provides a relevant understanding of the future use of the system within its business context</td>
<td>Validation and refinement of the actability concept</td>
</tr>
</tbody>
</table>
As discussed in Chapter 3, the focus of this dissertation is on actability and its operationalizations. Therefore, the main focus with respect to actability design is on the actability design method VIBA as such, and hence on the level of approach. The way to develop the approach and to learn more about it and about actability has been to use it to specify systems. As a consequence, the discussions in the remainder of the dissertation will focus on VIBA and its applicability, not on specific systems and work practices. However, as part of the research context these will be used as illustrations and points of reference.

As indicated above, the empirical work has produced output of two different kinds. Figure 8-3 illustrates these kinds of empirical output.

![Diagram](image)

**Figure 8-3:** The Empirical work as generator of ideas and data.

The first kind of output, referred to as **ideas**, represents reflections made during the empirical work that has been fed back to a continuous design process, and immediately resulted in changed ‘versions’ of VIBA and actability. The other kind of output, referred to as **data**, represents more systematically treated data, gained through observations and interviews. Although ideas, by nature, are more volatile than data, some reasons underlying new ideas have been documented in logbooks and hence constitute data. The dashed arrow from ‘result’ to ‘design’ indicates future re-design based on the results.

Altogether, the basic approach during the empirical work has been to (1) continuously improve VIBA (and actability) and (2) collect data of different kinds (see section 8.5) and do ‘formal’ analyses to try to understand what worked well and what did not, and why. It is important, though, to realize that the results are not generalizable, in a traditional sense, and that more empirical work is still needed.

As discussed in Chapter 3, the empirical work has been performed according to a qualitative research tradition. The reason for this is derived from the aims of the studies. The first applications (Applications 0 and 1) were, in essence, explorative studies, and the choice between quantitative and qualitative approach was, more or less, given from the start. Information systems development methods are complex phenomena, a method under development and continuous change even more so. Such circumstances require closeness between the observer and the observed and the circumstances made it difficult to specify expected results in advance (see Chapter 3.3. Hence, Application 1 was performed as action research in a minimal setting. During Application 2 we wanted to also involve other people to actively work with the method proposal, in order
to get some kind of ‘second opinion’. This led to Application 2 being carried out as a mixture of action research and observations. We (the researchers) created a specification of an IS which we handed over to an IT consultant firm for realization. In this way we could continue to refine VIBA while doing the requirements elicitation and analysis. And we could explore whether the consultants understood and appreciated the resulting specification. There was a big risk involved in this research design. The systems development had to follow a waterfall process, at least from analysis to design, which seemed to be in contradiction to our rapid development aim. Anyway, we decided to take that risk since it was also an opportunity to test if the documentation were communicable, in the extreme. During application 4, this idea of involving other people was developed even further. During this work, researchers that had not been involved in the development of VIBA worked together with one of the researchers that participated in Application 2. Their work was then followed up through interviews and analysed by this author.

Our interest in Application 1 was the specification. We had not planned to study the following phases due to uncertainties as to whether financial support for realization was available. As it turned out, the project was cancelled (or at least put to rest) even before the specification was fully completed, because of lack of money. However, we were able to collect quite a lot of data during the application and the fact that it was never completed did not affect the appropriateness of its inclusion in this study.

Due to circumstances beyond our control, the IS constructed during Application 2 was never implemented in the host organization. Otherwise we had planned to follow up the implementation to see whether the action orientation actually became visible in the final product. This important issue was elaborated during Application 4 and has been reported by Cronholm and Goldkuhl (2002).

In the following sections the four Applications are presented followed by a discussion of how data collection and analysis have been performed.

8.4 About the Actability Design Applications

This subsection gives a brief description of the four applications. The purpose of the descriptions is to give a background, and a ‘feeling’ for the different development efforts and their contexts.

8.4.1 Application 1: Production Planning in Mechanical Engineering

The first application was carried out at a Swedish mechanical engineering firm supplying parts to other companies (contract manufacturing). The firm was a small company with about 15 employees and no formal organization chart. Instead the managing director (MD), who was also the owner, handled sales, production planning, and all other managerial tasks. As it turned out, the MD, whose workload was continuously increasing, felt that some computerized tool was needed to ease the burden. The main problem was to match production schedules with customer demands and to reach higher precision in predicting when a particular product could be delivered. The goal was to create an information system that kept track of production schedules, and that could be used interactively to calculate delivery dates while still talking to a customer. At that time, this was impossible due to a number of factors. First, the MD was not always updated on the actual state of the production process. Second, everything needed to calculate

---

38 By ‘realization’ we refer to the design, construction, and implementation of an IS.
production times had to be memorized and calculations were performed manually. Some information was not even possible to retrieve instantly. Instead the MD had to ask some shop-floor workers about current capabilities. The system thus aimed to cover the sales process from negotiation, via order, to fulfilment and delivery. Hence it was a combined ‘sales support system’ and a ‘production planning system’. Furthermore, it was planned to replace current systems for customer administration and delivery administration. It was also supposed to co-operate with the administrative systems used for traditional bookkeeping. However, the production planning was at the heart of the planned system, and the key to the success of its other parts.

8.4.2 Application 2: Material Administration in Paper Production

The second application was carried out at a Swedish paper mill producing paper tissue from pulp, as well as different laminate products, of which tissue is an important ingredient. The IS to be developed was a quite ‘standard’ material administration system. The goal was to keep track of products and product categories, raw material, warehousing, suppliers, supplier orders, and deliveries of raw material. One part of the system was aimed at helping the production planner to create production schedules. Another part was aimed at helping the buyer to create supplier orders, based on the need implied by the scheduled production. A third part was designed for shop-floor workers to register deliveries and raw material usage. At that time, the buying and planning activities were supported by stand-alone spreadsheet and document processing applications. The warehousing was performed and maintained manually. However, much of the data needed for the new IS was already contained in different databases used for statistical and quality assurance purposes. So, there was a need to integrate the new system with a host of legacy applications and databases.

8.4.3 Application 3: Teaching Experiences

The third application was carried out at a University in the form of an intermediate systems development course, given five times through 2000–2002. This author and another lecturer performed the actual teaching. During the course, students were asked to create a complete specification of an information system by using one of several methods presented to them, of which VIBA was one. Approximately 20 system specifications have been made using the method. Additionally, VIBA and actability was included in a final written examination, so all students had to learn its basics.

8.4.4 Application 4: Planning in Home Care

This application was performed as a systems design project in a home care unit serving elderly people in a medium-sized Swedish local authority. The business can be characterized as complex with a high communication demands. The major task of Swedish home care is to help the elderly with daily routines, such as hygiene, cleaning, laundry, and shopping.

Each client has individual needs and a business goal is to maximize the individualization of care. The staff consisted of two home care managers who where responsible for the care unit and a number of home care assistants responsible for the day-to-day work with the elderly. In the ISD project four researchers, two home care assistants and two home care managers participated. The author of this dissertation interviewed three of the researchers to get their opinion of VIBA (see Section 8.5).
8.5 Actability Design Data collection

8.5.1 Selection of Applications
Application 1, 2 and 4 were selected ‘at random’ in the sense that they were all ‘stumbled across’ and, after evaluation, seemed to match our purposes. All the cases represented the kind of problem domains that we were addressing: business supporting information systems where communication problems were obvious, and where a business process-oriented approach was straightforward. Another approach might have been to actively search for appropriate possible applications, but we deemed that there was no need for that. The different applications represent widely varying types of information systems as well as development and usage context, which made it possible to try the method in different settings.

Application 3 was a natural choice since the method was already included in the information systems curriculum and it was a good opportunity to gain further experiences. Of course, this was a more controlled situation since we, as teachers, could stipulate the conditions for the students’ accomplishment. At the same time, ‘real life demands’ are hard to simulate convincingly, which is a shortcoming in this kind of research setting.

8.5.2 Records
The collection and recording of data focused primarily on positive and negative aspects of the systems development work and of the forms of documentation suggested by the re-designed VIBA. The sources of data, i.e. the (roles of the) actors participating in the applications, are summarized in Table 8-2.

<table>
<thead>
<tr>
<th>Roles</th>
<th>Application</th>
<th>Referred to as</th>
</tr>
</thead>
<tbody>
<tr>
<td>Researcher, requirements analyst, lecturer (the author)</td>
<td>1, 2 and 3</td>
<td>R1</td>
</tr>
<tr>
<td>Researcher, requirements analyst</td>
<td>Part of 1</td>
<td>R2</td>
</tr>
<tr>
<td>Researcher, requirements analyst</td>
<td>2 and 4</td>
<td>R3</td>
</tr>
<tr>
<td>Systems co-ordinator</td>
<td>2</td>
<td>SC</td>
</tr>
<tr>
<td>IT consultant, project leader and designer</td>
<td>2</td>
<td>C1</td>
</tr>
<tr>
<td>IT consultant, designer and programmer</td>
<td>2</td>
<td>C2</td>
</tr>
<tr>
<td>Requirements analysts, students</td>
<td>3</td>
<td>Students</td>
</tr>
<tr>
<td>Lecturer</td>
<td>3</td>
<td>L1</td>
</tr>
<tr>
<td>Requirements analyst</td>
<td>4</td>
<td>R4</td>
</tr>
<tr>
<td>Designer and programmer</td>
<td>4</td>
<td>R5</td>
</tr>
</tbody>
</table>

The studies can be characterized as combinations of action research and formative evaluation, using the terminology of Patton (1990). In this terminology action research is focused on solving some problems within an organization and formative evaluations are used to improve, for example, a program or product. Hence, the applications led to
results on two levels: information systems (specifications) for the participating organizations, and improved method and theory as primary research result.

Data collection was performed throughout the applications and recorded in research logs. Logs were also maintained by the IT consultants (C1 and C2) during their realization work in Application 2. The system co-ordinator at the paper mill (SC) also documented his experiences during the realization in a log. In addition to the logs, C1 and C2 were asked to fill out forms with comments during their work. The forms had pre-specified columns addressing positive and negative aspects of each type of document in the specification. Furthermore, the forms contained two pages addressing ‘missing information’ and ‘non-useful information’ in the specification that they might occasionally find during the work.

A total of seven interviews have been conducted, four during Application 2 and three during Application 4. The first two interviews were performed with the IT consultants after approximately 50% of the realization had been performed. As suggested by Patton (1990), the interviews followed an interview guide, specifying the topics and issues to be penetrated; see Figure 8-4.

<table>
<thead>
<tr>
<th>General</th>
<th>Opinion</th>
<th>How did that feel</th>
<th>Evolution over time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Work tasks</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Overall judgement</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Understanding of the business</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Understanding of the system</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Guiding/freedom</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Documentation</th>
<th>How used</th>
<th>Why</th>
<th>Positive</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action Diagrams</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Document Definitions</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use-Situation List</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I-Tables</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Prototypes</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>State Charts</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class Diagrams</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class Definitions</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Other documentation</th>
<th>How would it have been used</th>
<th>Why</th>
<th>How it was done instead</th>
<th>Opinion (what to do)</th>
</tr>
</thead>
<tbody>
<tr>
<td>…filled out during interviews…</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 8-4:** Interview guide (author’s translation from Swedish).

The interview guide was divided into three parts where each part was formed as a matrix of headings (y-axis) and dimensions (x-axis) from which to discuss the topic
suggested by the heading. During the interviews each ‘cell’ was marked when its area had been discussed.

The two last parts of the interview guide mirrored the forms discussed previously, but allowed for greater flexibility. The forms represent a standard open-ended inquiry whilst the interviews allowed for more reflection and follow-up questions. In accordance with the recommendations of Patton (1990), the interview guide was used as a check-list and did not direct the ordering of questions, et cetera. Instead, the interviews were performed more like conversations, allowing for different twists and turns to occur. These first two interviews were recorded on tape.

The third and fourth interviews were performed as unstructured open-ended interviews. That is, no interview guides were used, and the respondents were allowed to speak freely and reflect upon the work performed. This style of interview was chosen to promote a freer atmosphere in which respondents did not feel restricted but could reflect more unreservedly. During these interviews notes were taken. The three interviews (with R3, R4 and R5) performed during Application 4 took place after the specification of the system was finished and a working prototype had been delivered. These interviews followed the same pattern as the first two and utilized an interview guide similar to that shown in Figure 8-4. During Application 3, a log was maintained containing ideas and suggestions from L1 as well as from students.

A summary of the different records is shown in Table 8-3.

Table 8-3: Summary of collected Actability Design data and its origin.

<table>
<thead>
<tr>
<th>What</th>
<th>Source</th>
<th>When</th>
<th>Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>Research log 1 (RL1)</td>
<td>R1</td>
<td>Application 1.</td>
<td>Written chronologically.</td>
</tr>
<tr>
<td>Research log 2 (RL2)</td>
<td>R1</td>
<td>Application 2,</td>
<td>Written chronologically.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>specification.</td>
<td></td>
</tr>
<tr>
<td>Research log 3 (RL3)</td>
<td>R1</td>
<td>Application 2,</td>
<td>Written chronologically.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>realization.</td>
<td></td>
</tr>
<tr>
<td>Consultant log 1 (CL1)</td>
<td>C1</td>
<td>Application 2,</td>
<td>Written chronologically.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>realization.</td>
<td></td>
</tr>
<tr>
<td>Consultant log 2 (CL2)</td>
<td>C2</td>
<td>Application 2,</td>
<td>Written chronologically.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>realization.</td>
<td></td>
</tr>
<tr>
<td>Interview 1 (I1)</td>
<td>C1</td>
<td>Application 2.</td>
<td>Transcription of recorded interview performed when approximately 50% of the realization was completed.</td>
</tr>
<tr>
<td>Interview 2 (I2)</td>
<td>C2</td>
<td>Application 2.</td>
<td>Transcription of recorded interview performed when approximately 50% of the realization was completed.</td>
</tr>
<tr>
<td>Written comments 1 (WC1)</td>
<td>C1</td>
<td>Application 2.</td>
<td>Submitted during I1, pre-structured form.</td>
</tr>
<tr>
<td>Written comments 2 (WC2)</td>
<td>C2</td>
<td>Application 2.</td>
<td>Submitted during I2, pre-structured form.</td>
</tr>
<tr>
<td>Interview 3 (I3)</td>
<td>C1 and C2</td>
<td>Application 2.</td>
<td>Written notes from interview performed after the realization.</td>
</tr>
<tr>
<td>Interview 4 (I4)</td>
<td>SC</td>
<td>Application 2.</td>
<td>Written notes from interview performed after the realization.</td>
</tr>
</tbody>
</table>
8.6 Actability Design Data Analysis

Analysis of qualitative data can be performed in several different ways. At first, the Grounded Theory approach by Strauss and Corbin (1998) was planned to be used as a source for structuring the work. However, after some initial analysis (open coding and part of axial coding) it was realized that Grounded Theory was not an optimal approach in this case. The reason was that the analysis tended to be too broad. One assumption of Grounded Theory is that the analyst should be as open-minded as possible and strive not to pre-categorize. Instead categories are to be built up, and related to each other, based on empirical data. The interest in the data was thus too focused to actually fit within the Grounded Theory framework. In fact, much of the data was inherently pre-categorized and semi-structured, such as the forms with positive and negative aspects of document types, and the interview guides. The research interest was not in generating new categories but in properties of already existing ones. Therefore an approach that made use of the structures already created was favoured. This was following Patton (1990, p. 376), who states that ‘answers from different people can be grouped by topics from the guide… The interview guide actually constitutes a descriptive analytical framework for analysis.’ When the data had been structured in this way, patterns in the answers and generalizable and reoccurring ideas were sought. It is important to observe that in the analysis the author’s own experience and fellow researchers’ experiences during the action research are treated as data. This implies a great risk of bias towards this author’s personal opinions and beliefs. This is, however, a risk in all interpretative studies, and the sources have been kept separate throughout analysis in order to make the results traceable.

It is important to be aware that after the specification was handed over to the consultants during Application 2, as well as after the interviews, some rounds in the ‘ideas loop’ of Figure 8-3, p. 137, had been performed, partly as a consequence of the fact that Application 1 was still ongoing. Similarly, the version of VIBA presented in this dissertation deviates from that which was the basis for the interviews during Application 4. Actually, there have been five major externalized versions of VIBA as shown in Table 8-4.

In essence, this means that the results of the analyses do not refer directly to the latest versions of VIBA (and actability). Furthermore, due to the transient state of VIBA, no particular externalized version is considered a basis for Application 1. In-

---

39 Broad in the sense of ‘unfocused’. Of course, interesting information was revealed, but much of that information was outside of the scope of this dissertation.
stead, VIBA’93, modified with ideas from Application 0 and Application 1, forms VIBA’98.

**Table 8-4**: Different externalized versions of VIBA.

<table>
<thead>
<tr>
<th>Version</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIBA’93</td>
<td>Point of departure for the work.</td>
</tr>
<tr>
<td>VIBA’98</td>
<td>Basis for Application 2.</td>
</tr>
<tr>
<td>VIBA’99</td>
<td>Reported by Ågerfalk (1999b) and Ågerfalk and Goldkuhl (2001). Basis for Application 3.</td>
</tr>
<tr>
<td>VIBA’00</td>
<td>Basis for Application 4. Used during Application 3.</td>
</tr>
<tr>
<td>VIBA’02</td>
<td>Reported in this dissertation.</td>
</tr>
</tbody>
</table>
Chapter 9

Versatile Information and Business Analysis


9.1 Chapter Outline

In this chapter the systems development method VIBA is described. The chapter starts with a discussion about what VIBA is and what it is not. Then the high-level structure, or framework, of the method is presented, followed by descriptions of its main areas of interest (its focal areas) and its constituent parts. The chapter ends with a presentation of the method’s underlying conceptual model.

9.2 What VIBA Is and Is Not

Methods are used during systems development to guide the development process. Methods suggest certain activities to be performed. The outcome of a systems development process is, of course, one or several software systems or system components. In addition to the final piece of software, methods usually recommend certain deliverables to be produced along the way.

One important example of deliverables is different models of the system and its environment. These models can be thought of as blueprints used in subsequent steps in the development process. Models are expressed in a more or less formalized notation. Class models are, for example, used during analysis in order to describe requirements on what information the system shall contain. The elements of the models represent the concepts used to talk about the problem domain, for example classes. The concepts

40 Versatile Information and Business Analysis.
used within a method direct developers’ attention to certain kinds of phenomena in the problem domain. This is another important task of a method: to direct developers’ attention towards relevant phenomena and tasks. In the case of VIBA, the intention is to direct attention towards aspects important for the actability of the IS being developed.

VIBA is not a ‘completely new’ or radically different method. Rather, it aims to reuse as much current knowledge and good practice as possible; VIBA makes use of existing concepts, activities and notations and adds some where necessary.

VIBA is mainly focused on requirements. Using the method results in a detailed requirements specification including:

- functional and non-functional requirements,
- requirements on information content, and
- user interface design.

VIBA suggests analysing interaction between humans and IT-systems on a ‘per use-situation basis’ as an integral part of business modelling. This is necessary in order to cope with complexity (by analysing the parts of the system separately) and still maintain a holistic view of the business context, in which the interaction is taking place, i.e. part of bridging the RE-gap (see Section 1.2.2). Furthermore, it is suggested to perform the analysis of each interactive use-situation (ISP) on a ‘per ae-message basis’, for similar reasons. In essence, Actability Design according to VIBA is concerned with three questions:

- Which ae-messages have to be formulated and sent to achieve the desired business effects? That is, what e-actions have to be performed to produce the desired output?
- For each ae-message: who is the communicator, performer, and intended interpreter, and what is the propositional content, communication function, and communication effect?
- What computer support is required to formulate and send the ae-messages, and what information from previously sent ae-messages is required to formulate the one at hand?

The questions concern three aspects of ae-messages (which ae-messages are relevant for the business context at hand; what constitutes each relevant ae-message; and what is needed, in terms of previously sent ae-messages and computer support, to formulate each relevant ae-message?).

However, requirements alone do not construct a system. VIBA is intended for construction of software systems and related manual routines. In most cases, it is unrealistic to first try to specify all requirements and then construct the system following a so-called waterfall process model. Rather, requirements should be allowed to evolve during the whole development process.

VIBA is intended for use with an incremental and iterative process model. With such a model, a small part of the system is built initially, containing only the most critical requirements. ‘Most critical’ refers to the requirements representing the most value for the users and the one most likely to be problematic to implement. This system is then extended in increments until the system is completed. This is in line with, for

---

ISP is an acronym for Interactive use-Situation Proposal. That is, it is considered as preliminary until the system is delivered. Until then its limits are allowed to change, and they usually do.
example, Boehm’s (1988) spiral model and the Rational Unified Process (RUP) (Kruchten, 1999). There is clearly a need to align VIBA with other methods and techniques for the actual construction of software.

9.3 Focal Areas of VIBA

VIBA takes the business and its actors as the point of departure for systems development. VIBA also promotes a process-oriented action-perspective on doing business. This means that systems development starts with understanding what the new system is going to be used for, and who are going to use it. VIBA is not about building a model of reality to store in a database. VIBA is about recognizing that human actors as well as computers perform important actions by use of information technology.

VIBA is about action. The concept of social action is what drives systems development forward. It is from understanding what actions are going to be performed by and through the system that requirements are derived.

Introducing new IT-based solutions in a business will always affect the way business is done. Sometimes in non-radical ways, sometimes the whole underpinning of the business relies on the new solution. Hence, systems development is business development. By being aware of this and taking the business both as point of departure and as final destination, a solid foundation for developing systems that meet the business’ needs is laid.

During systems development, different phenomena (or concepts) are focused on at different times. Such concepts usually correspond to the modelling primitives in the various models produced (See Section 2.3). In VIBA, a coherent set of concepts that are focused together is referred to as a focal area of the method. A focal area is a unit of interest where the focused concepts are closely related to each other and where it is not possible, or at least not viable, to study one concept of the focal area in isolation from the others. A focal area thus ‘highlights’ some aspects of the piece of reality under scrutiny. This piece of reality should usually not be considered in isolation. The focused area must always be understood in the light of the other areas that are, for the moment, de-emphasized, in a dialectic manner. Furthermore, insights gained from studying one area are typically used as input (or background knowledge) when focusing other areas subsequently.

In VIBA there are seven main focal areas: Goals, Problems, Activities, Actions and Messages, Interaction, Documents, and Concepts.

VIBA begins with a thorough analysis of the business. During this analysis the business’ goals and problems are analysed. A vision of the new system is created and documented and different prerequisites are stated explicitly. The reason for starting with goals and problems is to steer the development project in the right direction.

When an understanding of this development baseline has been established it is time to dig deeper into the activity structures of the business – who does what to whom, how, when and why, by use of information technology? When analysing the activity structures, it is important to carefully allocate activities to different performers. Some activities may be performed solely by human actors, others by human actors in interaction with IT-systems. Additionally, and this is important, some activities may be

---

42 Note that the term ‘activity’ is used differently here than its use in Activity Theory (Kuutti, 1996; Engeström, 2000). In VIBA (and thus actability) terminology, an activity may correspond to several ‘Activity Theory’ activities.
performed by IT-systems alone. We may choose to let the system perform certain activities on the behalf of the human actor responsible for it. The system, in this case, acts as an (more or less intelligent) agent (see Chapter 4).

Business processes are made up of activities. Activities result in either material objects, such as products and deliveries, or communicative objects, such as commitments and contracts.

By analysing the activities of the business and how they are structured, an understanding of ‘what is going on’ is established. The understanding is still at a fairly high level so far, and we need to dig deeper to understand the requirements on the IT-systems that are going to be a part of the action in the business.

Within VIBA, activities are thought of as aggregates of e-actions. To understand system requirements, we need to understand the individual actions that form the activities of the business. Actions performed via IT-systems are performed by use of language. Therefore we are primarily concerned with communication action at this level of the analysis. The language used is the professional language used by the business’ actors in their daily work. The result of e-actions is ae-messages sent from the performer of the actions to one or several interpreters. The language used in performing the actions, and hence in the resulting messages, can be analysed in terms of the concepts of which they are build up and the relationships between those concepts.

Actions can be performed via the IT-system and therefore the interaction between humans and computers is vital for the successful performance of actions. Actually, the system can aid the user in formulating the utterance that he or she is going to communicate through the system, and even in doing the actual uttering. Thus, the design of the human-IS interaction is supported by VIBA based on the same social action-perspective that runs throughout the whole method.

Human interaction with an IS is performed by use of interactive documents – forms, windows, screens, et cetera. These documents are thought of as an action medium – an arena for playing an interactive language game. There are also other kinds of documents. The most important being the static documents that is output from the system. Such documents are used primarily as carriers of the messages communicated through the system and serve as foundation for actions outside of the system. An example would be a supplier order printed and sent to a supplier. The supplier then acts in response to the message (the order) carried by the document and, hopefully, delivers the requested goods.

As we can see there are three levels of concern within VIBA: the activity level, the action level, and the interaction level. All these three levels must be understood in order to construct a system that supports the business’ actors in performing their actions and activities in interaction with it.

9.4 High-Level Structure of VIBA

As stated above, VIBA is a method for specifying requirements on information systems and businesses to be developed. In order to bridge the RE-gap (see Section 1.2.1), information systems development is considered to be a special kind of business development. The aim is to build actable information systems since the method builds explicitly on the concept of actability. One aim of the method is to facilitate a rapid development setting and to make use of rapid application development tools. Another aim is to deliver specifications that enable traceability from detailed systems specifications
to business models and vice versa. The method context is business information systems in process-oriented organizations.

VIBA adopts three interacting approaches: modelling, prototyping and evaluation. Modelling refers to the analytical creation of models of the system and its environment. Prototyping, on the other hand, concerns experimental activities during which examples of the system are created. Evaluation is based either on analytical models or developed prototypes. It is important to notice that these three parts do not constitute separate phases or workflows (defined below) within VIBA. Instead, they should be understood as representing an attitude towards systems development that VIBA promotes. Specifically the evaluation part is implicit in the proposed ways of working. The idea is to promote a reflective approach that constantly evaluates models and prototypes. Evaluation means to critically examine the specification and verify its internal consistency, as well as external validity with respect to the business being designed and the users. The key concept for evaluation is hence ‘why’ – why do the models and prototypes look the way they do?

The framework or ‘high-level structure’ of VIBA can be understood as a combination of four parts:

Â Three workflows.
Â Three developer roles and two client roles referred to as workers.
Â 21 types of documents to be produced referred to as deliverables.
Â A suggested development process model to guide the development activities.

The framework of the RUP influences that of VIBA. Therefore, some terminology and visual representations from the description of RUP has been adopted so that people familiar with it should feel comfortable with the description of VIBA. It is also the intention that VIBA should be possible to use as a complement to RUP and to other processes in order to acknowledge the actability aspects of information systems.

In the remainder of this section the four components of the VIBA framework are presented, starting with the three workflows.

9.4.1 Workflows

A workflow can be thought of as a partially ordered set of activities to perform during development. A workflow is typically concerned with one or several focal areas. Based on the three levels of concern mentioned above, VIBA consists of three different workflows:

Â Process Modelling
Â Action Modelling
Â Interaction Modelling

During Process Modelling the focus is on the activity structures of the business of which the new system is to be a part and on different prerequisites that set the scope for the system. Attention is paid to an overall vision of the new system as well as to business goals and problems that the system is supposed to solve. During Action Modelling the focus is on how the system is going to be used from a business perspective. The central concept is that of the use-situation. Note that ‘use’ should be interpreted as an abstract concept during Action Modelling. Interaction Modelling draws attention to actual use in terms of human-IS interaction. During Interaction Modelling attention is turned towards the actual properties of the system (external properties as well as inter-
nal). How is the human-IS interaction going to be realized and how is information going to be structured?

9.4.2 Workers

A worker\(^\text{43}\) is a role played by an individual participating in a development project. The worker concept can be thought of as a hat. One person can have several hats but will only be wearing one hat at a time. Workers are either developers or clients. Developers are working in a project for the software developing organization. Clients represent the business for which the system is built. The three developer workers suggested by VIBA are the Business Analyst, the Systems Analyst and the User Interface Designer. The two client workers are the System Owner and the End User.

A Business Analyst should be a person with knowledge about business in general and preferably the kind of business that the new system is going to be a part of. Such a worker must also be familiar with how information technology can be used to deliver something of potential value to the business’ clients. Ultimately a business analyst is a businessperson with a strong interest in technology. The Business Analyst is primarily responsible for the workflow Process Modelling.

The role of a Systems Analyst is to elicit, analyse and document requirements. A Systems Analyst should be familiar with both business and computers. However, the most important characteristic is an ability to realize what can actually be done with a computer – today and tomorrow – and to some extent how that should be done. The Systems Analyst is primarily responsible for the workflow Action Modelling and part of Interaction Modelling. Typically, people wearing this hat are involved in most development activities.

The User Interface Designer should be a person who understands the capabilities and limitations of humans interacting with computers. It is a person who combines creativity and analytical abilities – an artist with engineering skills. The User Interface Designer is primarily involved in the workflow Interaction Modelling but should preferably play an active role throughout the whole development process (cf. Göransson (2001) on the developer role Usability Designer, which approximates the role of the User Interface Designer in VIBA).

As the name suggests, an End User is a person that is going to use the system. This includes the communicators and performers of actions as well as interpreters (see Section 4.5.1). The System Owner is the one responsible for the system and usually the one that pays for its development. These two types of clients might coincide, but not necessarily.

Each worker (especially developers) performs specific activities prescribed by VIBA. Further, for each deliverable (see below) there is a responsible worker. However, most of the activities are performed in co-operation between several people representing different workers, even though there should always be one single person ultimately responsible.

9.4.3 Deliverables

During systems development different kinds of deliverables are produced. These include both software components (source code and executables) and systems document-
tation. Systems documentation is produced to document the different results of analysis and design. These serve both as input to subsequent activities and as an important rationale for the construction of the system. The latter aspect is important for maintenance of the system in the future. The documentation reflects how the developers conceptualized the system during development and helps understanding how the different parts relate.

The VIBA documentation consists of different diagrams and textual documents (which might be more or less formalized), such as prosaic documents, lists, tables, *et cetera*. These are grouped into five deliverable sets: the Prerequisite Set, the Activity Set, the Interaction Set, and the Conceptual Set. The different deliverables contained in the different deliverable sets are described in detail in Sections 9.5 to 9.7, but in order to bring about an initial understanding they are presented briefly already at this point. In Section 9.8 a more elaborated overview of the different VIBA deliverables is given, which explicitly relates them to each other.

The Prerequisite Set contains the identified prerequisites for the development and the ‘high-level requirements’, such as business goals to be met, which set the scope for the system. It contains the following types of deliverables:

- Business Definition
- Vision Document
- Problem List
- Problem Diagram
- Goal List
- Goal Diagram
- Supplementary Prerequisites

The Activity Set documents the activity structures of the business. This includes activities and information flows, inputs and outputs to activities and performers. The Activity Set contains the following types of deliverables:

- Action Diagram
- Role Definition
- Role Diagram

The Action Set contains descriptions of what actions are to be supported by the system and how these are structured into use-situations. It contains the following types of deliverables:

- Message Definition
- Use-Situation List
- Use-Situation Diagram

The Interaction Set documents how the interactive and consequential use-situations are going to be realized as sequences of human-IS interactions. It contains the following types of deliverables:

- e-Interaction List
- Document Definition
- Document Prototype
- Document Statechart
The Conceptual Set documents the concepts used in the messages carried and stored by the system, how these are related and how they change over time. It contains the following types of deliverables:

- Glossary
- Class Definition
- Class Diagram
- Class Statechart

In addition to these deliverables, there are usually other important things to consider as input to the development. In VIBA, these are collectively referred to as patterns, which play an important role in all three workflows. ‘Patterns’ refers to pre-existing solutions to problems similar to the ones at hand. Patterns might be requirements patterns (Maiden et al., 1998), analysis patterns (Fowler, 1997), or design patterns (Gamma et al., 1995), that is, in accordance with the traditional use of the term in the fields of information systems and software engineering. Moreover, existing artefacts, such as already implemented information systems and businesses, might also constitute patterns. The real issue of highlighting patterns as an important part of VIBA is to encourage an attitude towards information systems development that is open-minded and that counteracts the so-called ‘not invented here syndrome’.

9.4.4 Overall Structure of VIBA

So, now we have three workflows, three developer workers (roles), two client workers (roles) and five deliverable sets – and the seven focal areas from Section 9.3. Now it is time to see how all these relate in an overall high-level structure of VIBA. Figure 9-1 depicts this structure.

From Figure 9-1 we see that Process Modelling consists of a set of activities performed by a Business Analyst worker. The activities are performed in close collaboration with the two client workers, the System Owner and the End User. The outcome of the activities forms two deliverable sets, the Prerequisite Set and the Activity Set. These two sets are used in the other two workflows and hence there is an implicit ordering among the workflows. We start with Process Modelling, continue with Action Modelling and finish with Interaction Modelling. However, as is discussed in the next section, the different workflows are returned to several times during development.

During the workflow Action Modelling, a Systems Analyst worker in co-operation with End Users uses the Activity Set to produce the Action Set.

The Action Set constitutes the main input to Interaction Modelling (together with parts of the Prerequisite Set). During Interaction Modelling a User Interface Designer works on the Interaction Set and a Systems Analyst Worker on the Conceptual Set, both in co-operation with one or several End Users.
9.4.5 Modelling and Validation of Actability Requirements

As mentioned above, use-situation proposals are derived explicitly from business modelling. These can then be used as a basis for a more detailed analysis concerning the interactive actions and the design of interactive documents. The ‘performer part’ of interactive actions is thus focused. We turn our attention to the information that flows (the messages sent) between the actor and the system, its origin and form, to analyse functional requirements and requirements concerning user interface design in the context of business processes, i.e. actability requirements. Analysis of humans’ interaction with IT-systems for actability is preferably performed with a mixed analytical and experimental approach in a dialectic manner (Mathiassen et al., 1995). Analytical models are continuously validated against document prototypes and prototypes are continu-
ously verified against analytical models. This way both consistence and relevance are achieved, which is difficult to obtain by using only one of the approaches exclusively.

Both modelling and validation of requirements can be performed based on the ‘three questions of Actability Design’ as discussed above. The ae-messages to be sent in each use-situation can be identified by looking at the results of the corresponding action(s) in the business model. These results are the means to reach desired effects of the use-situation. For each ae-message there should be a propositional content and an action mode that support those effects. Furthermore, there should be possible e-interactions and initial IS actions that permit, promote and facilitate the formulation of those ae-messages.

During this process both the contents and demarcations of the ISPs are likely to change. Results from one of the case studies (see Chapter 10) show that earlier assumptions about the business action structure might have to be reconsidered and the analysis of interaction thus becomes a valuable source of information also for business process modelling. This is an example of the need to integrate business modelling and interaction design. The necessity to formalize the action structure reveals information which is hard to elicit only by interviewing and observing actors. By putting actors in front of the future system they feel the need to be explicit, and they instantly see the effects of alternative business descriptions.

**9.4.6 Development Process Model**

A development process model dictates the overall approach to be used when applying a method – how and in what order the activities of the method are to be carried out. The suggested development process model of VIBA can be characterized as being incremental and iterative. Incremental means that projects start small and expand the scope in increments. For example a first version of a system with only the most vital action potential might be constructed and delivered. This system is then extended with added functionality. Iterative development means that a development project is divided into several iterations constituting ‘mini projects’.

Above the three workflows of VIBA were introduced. A workflow should not be confused with a development phase. Which phase a project is in depends on how far the project has proceeded. With a traditional waterfall process model it is common to talk about phases in terms of analysis, design and implementation. Hence, it is easy to jump to the conclusion that the workflows of VIBA correspond to such phases. However, a workflow is not a phase. It is a collection of activities to perform during development. Each workflow might be addressed during each and every phase of development, and usually is! In the development process model of VIBA we talk about four phases (adopted from the RUP): inception, elaboration, construction, and transition.

During the first phase – the inception phase – the purpose is to establish the scope of the system and to identify core requirements in order to understand what is going to be built. During the second phase – the elaboration phase – the system requirements get more and more detailed. After this phase we feel confident about what the system is going to be like. The third phase – the construction phase – is where the major part of the system is constructed. The final phase – the transition phase – is where the system is ‘delivered’ and installed on the customer site. Figure 9-2 shows the relationship between workflows and phases.

As shown in Figure 9-2, each phase consists of one or several iterations. Remember that an iteration is like a ‘mini project’ in itself. This means that during each itera-
All workflows should (or at least could) be considered. However, the emphasis put into the different workflows differs depending on what phase the project is in. During the inception phase, the main emphasis is on process and action modelling. During elaboration more effort is put into interaction modelling.

<table>
<thead>
<tr>
<th>Phases</th>
<th>Inception</th>
<th>Elaboration</th>
<th>Construction</th>
<th>Transition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workflow</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Process Modelling</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Action Modelling</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interaction Modelling</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 9-2:** The development process model of VIBA.

VIBA is focused on requirements but there are other aspects to focus on in order to construct a system. These aspects would typically be emphasized during the elaboration and construction phases. However, VIBA can still be used as a stand-alone method for requirements analysis. This might be useful if, for example, a requirements specification is to be matched with a specification of an off-the-shelf product such as a (standard) business system. It might also be useful if the development situation requires that a requirements specification is going to be frozen and handed over to someone else for construction. In that case, VIBA would typically end with the elaboration phase and no changes in the requirements would be allowed after that. This is usually not a recommended approach but it might very well be considered if the business and requirements are well understood from the outset.

Note that, as shown in Figure 9-1, p. 153, End Users are supposed to be actively participating in all three of the VIBA workflows. This is important in order to apply VIBA as a user-centred process, which is the intention. The international standard ISO 13407 (ISO 13407, 1999) identifies four key-principles for user-centred design:

- Appropriate allocation of functions between users and system
- Active involvement of users
- Iterations of design solutions
- Interdisciplinary design groups

These principles are important to take into consideration when working with VIBA. However, it is important not to overemphasize the interacting end user, but to always strive for a balance between the business and the individuals within the business – between the organizational approach and the cognitive approach (see Section 1.2.3).

### 9.5 Process Modelling

The aim of the workflow Process Modelling is to gain initial understanding of the actions performed by and through the information system under construction, that is, to
understand why and for whom the system is built. Another important issue is to explain different prerequisites for the construction.

Some development incentive always triggers the use of VIBA (or any other development method, for that matter). This incentive can be more or less well grounded and consists of a decision, or at least a belief, that a system is going to be built. In most cases, such a decision is preferably based on a feasibility study\textsuperscript{44}. However, if no such pre-study has been conducted, VIBA still provides enough support to succeed with the development.

Throughout Process Modelling a Glossary document is used to define the terminology used both within the business under investigation and internally within the project. This is an important document to keep alive so that everybody knows what everybody else means. A Systems Analyst is responsible for this document, which is also an important input to the activity Analyse Concepts during Interaction Modelling (see Section 9.7).

In addition to Systems Owners and End Users as information sources, various existing documents are an important input for Process Modelling. Examples of important existing documentation are different policy documents, standards and strategy documents, such as overall IT strategy and system structure documents.

Process Modelling consists of three activities, which are the responsibility of a Business Analyst: Analyse Prerequisites, Analyse Activity Structure and Analyse Performers.

### 9.5.1 Analyse Prerequisites

The reason for analysing prerequisites is to bring an understanding of the business under development, and to make that understanding inter-subjective among the different people involved (developers, clients and possibly other stakeholders). Analysis of prerequisites concerns the formulation of a Business Definition, the analysis of problems and goals, the development of a vision for the new system and the possible elicitation of supplementary prerequisites.

#### 9.5.1.1 Business Definition

A Business Definition document defines the results of the business (its products and services), what customers and clients the result is produced for, the main tasks that have to be carried out to create the results, and the groups of people involved in the process. Additionally, it also documents important prerequisites in terms of required raw material and suppliers, as well as financial backers.\textsuperscript{45}

The main purpose of the Business Definition is to ensure that developers do not ‘forget’ the business, its intended results used to satisfy customers, and the activities performed to create such satisfactions. Doing this early during systems development emphasizes that information systems are developed with the purpose of adding value to the business – and thus to the business’ customers (Goldkuhl, 1993).

\textsuperscript{44} Preferably, such as study should be performed by use of Change Analysis (Goldkuhl and Röstlinger, 1993) since that particular approach shares many concepts with VIBA.

\textsuperscript{45} The Business Definition goes hand-in-hand with Goldkuhl and Röstlinger’s (1999) notion of a practice, described in Chapter 15, of which a business is a special case.
9.5.1.2 Analyse Problems

Before development actually starts, it is important to gain understanding of the problems that the new system is supposed to solve. Thus, problems are to be identified, formulated and analysed.

Following the approach suggested by the Change Analysis method (Goldkuhl and Röstlinger, 1993), identified problems can be documented in a Problem List. Usually it is not enough to simply list problems, one needs to analyse how problems relate to each other as well. That is, to identify causes and effects of problems. The reason for undertaking such analysis is to structure problems in ‘problem hierarchies’ in order to understand which are the ‘real problems’, that is, central problems that are the source of all trouble; these are the problems that are most important to solve. The analysis of problems’ causes and effects can be documented in a Problem Diagram consisting of a directed graph where nodes represent problems and vertices represent causal relationships between problems. Note that a Problem Diagram graph must not necessarily be a rooted tree (it might in fact even be disconnected).

The most important problems to solve are those represented by nodes having in-degree equal to zero (sources), referred to as root problems. Nodes of a Problem Diagram graph having out-degree equal to zero (sinks) correspond to problems that are usually solved ‘on the fly’ when the harder problems (closer to root problems) are considered, given that their in-degree is greater than zero.

9.5.1.3 Analyse Goals

To develop (re-engineer) a business is (or at least ought to be) to move away from the problematic and approach the desired. Therefore it is not enough to analyse the problems. One also needs to examine the business’ goals. Goals might exist on different levels, from high-level goals such as strategies and policies to low-level goals such as norms and business rules for a particular business process or activity. During Process Modelling it is important to concentrate on the goals that specifically relate to the IS to be developed in order to keep focus on the task at hand, that is, systems development. Goals can be documented in a Goal List, similar to problems in the Problem List.

Similar to problems, goals have to be analysed in terms of causes and effects. This is to discover which goals are the most important to achieve in order to decide where to put most development resources. Furthermore, it is possible that some goals contradict each other. Such ‘goal conflicts’ are important to identify and to solve, if possible.

The analysis of goal causes, effects and conflicts can be documented in Goal Diagrams. A Goal Diagram consists of a directed graph with nodes representing goals and with two different kinds of vertices: one kind representing goal/sub-goal relationships, whilst the other represents goal conflicts. As a notational convention Goal Diagram graphs are drawn in the opposite direction of Problem Diagram graphs – having the most prominent goals at the top and sub-goals beneath. Hence, nodes with out-degree equal to zero are the most prominent (important to achieve) as opposite to Problem Diagrams.

As mentioned above (see Footnote 44, p. 156), VIBA should optimally be preceded by Change Analysis (CA). During CA the problems and goals of (the part of) the business that the new IS supports are treated as described above. The results of CA can then be used directly after filtering goals and problems that obviously do not relate to the IS being developed. Otherwise this part of the method (Analyse Problems and Analyse Goals) must be worked through more carefully.
9.5.1.4 Develop Vision
To develop a vision of the system to be built is to describe the primary features of the system, often in a metaphorical way. The vision should not be considered as a requirement, but rather as something to keep in mind and steer towards when eliciting and defining requirements. One important function of the vision is to gain agreement on what problems need to be solved and what goals are to be achieved. It is also important in order to identify the system’s stakeholders and to define its boundaries. A Vision Document is typically written in a prosaic form, possibly in conjunction with some variant of ‘rich pictures’ (Checkland, 1981).

9.5.1.5 Supplementary Prerequisites
In addition to the business per se and its problems and goals, there are in general a number of different prerequisites that delimit the development work but also give rise to different possibilities. It is very important to make such prerequisites explicit early on. Prerequisites might be of different kinds. Goldkuhl (1993) lists some examples, such as: strategic, organizational, hardware, software, security, legal, work environment, integration with other (legacy or new) systems, etc. Identified prerequisites are documented in a Supplementary Prerequisites document. The Supplementary Prerequisites document, the Problem List and the Goal List, are the most important tools in guiding the systems development process.

9.5.2 Analyse Activity Structure
As stated above, the method context of VIBA is business information systems within process-oriented organizations. A business process consists of activities ordered in a structured way with the purpose of producing valuable results for clients. Different persons within the organization (and even outside) can perform such activities. Customers must, for example, perform certain actions, such as inquiring and ordering. As discussed in Chapter 4, some actions of the organization can also be performed automatically by computerized information systems.

Modelling and designing business processes means describing different (types of) actions that should be performed within a business. The process logic should capture how different actions are related to each other and different ‘firing conditions’ for actions (that is, action triggers).

9.5.2.1 Actions and Activities in a Business
Actions can be classified as material actions or communication actions (see Chapter 4). When discussing the concept of e-action in Chapter 5, what was referred to was mainly the latter, that is, to communication actions. In a business firm many actions performed are however material actions, for example changing the location of some raw material in a warehouse. Note that in practice, an action might be both material and communicative. Moving the raw material might be done to explicitly state where raw material of a certain kind shall be kept, for example. Furthermore, when analysing business processes during a systems development effort it is usually of no interest to dig into subtle details of the actions performed. Therefore the term ‘activity’ is used in VIBA, which is thought of as an action aggregate of one or more actions – material or communicative. An activity might thus ‘produce’ one or more ae-messages as well as one or more interventions in the physical state of the business. The concept of e-action as a more fine-grained analytical tool is thus deferred till Action Modelling (see Section 9.6).
9.5.2.2 Identifying and Describing Activity Structures

There are several techniques available to describe business processes such as, for example, sequence diagrams of the Unified Modelling Language (UML) (Booch et al., 1999) and Data Flow Diagrams of Structured Analysis (Yourdon, 1989). Due to inadequate semantics in other approaches, Action Diagrams (Goldkuhl, 1992; 1993) are used in VIBA to describe business processes. Chapters 6 and 7 introduced Action Diagram notation briefly. In this section it is described in more detail.

Action Diagrams explicitly describe different activities of a business process and how these activities are related to each other. Actions performed by human actors as well as automatic actions are considered. Note that Action Diagrams describe types of activities, and not the actual occurrences. Action Diagrams can be used to describe material flow and information flow within business processes (see Table 9-1).

| Table 9-1: Basic notation used for activities, action objects and flows in Action Diagrams. |
|----------------------------------------|--------------------------------------------------|
| **Notation**                           | **Description**                                  |
| [ ]                                    | Information (oral or written) action object.     |
| [ ]                                    | Material action object.                          |
| [ ]                                    | Information flow (communication).                |
| [ ]                                    | Material flow.                                   |
| Activity [Performer]                   | Activity with named performer.                   |

Material and information (as action objects) are described and related to activities as prerequisites (input) or results (output). One important notational feature is that Action Diagrams describe the performer of each activity; i.e. what actor/actor group (actor role) or which IS that is supposed to perform the particular activity. Interactive activities might be one-way, [A → B], (from one performer to another) or two-way, [A ↔ B], (a dialogue between performers). Two-way interaction is most common but one-way interaction is typically the case with, for example, reports and orders. Additionally, activities can be interactive with several performers, which would be the case when, for example, a customer provides a salesperson with information entered interactively into a system, denoted as [Customer → Salesperson ↔ System].

In addition to the basic action objects of Table 9-1, Action Diagrams make it possible to, as shown in Table 9-2, distinguish between:

- Encapsulated information (i.e. information not directly readable by humans, in which case there is a need for a computer to translate and display it).
- Action object storage (used when activities add action objects to already existing quantities).
- Knowledge (i.e. not externalized information that exists within some human actors mind).
Table 9-2: Special notation used for encapsulated information, stored action objects, knowledge, and non-action in Action Diagrams.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Encapsulated information" /></td>
<td>Encapsulated information.</td>
</tr>
<tr>
<td><img src="image" alt="Store of action objects" /></td>
<td>Store of action objects. Applicable to both information and material.</td>
</tr>
<tr>
<td><img src="image" alt="Knowledge" /></td>
<td>Knowledge, which has not been externalized.</td>
</tr>
<tr>
<td><img src="image" alt="Inaction" /></td>
<td>Inaction, i.e. an action object that is the result of an omitted action.</td>
</tr>
</tbody>
</table>

In addition to performers, the physical location where an activity is to be performed can be shown in Action Diagrams by adding an asterisk and the name of the location after the performer list (see Table 9-3).

Table 9-3: Notation used to show several performers and activity location in Action Diagrams.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Delivery" /></td>
<td>Activity with several performers, named location and resulting action objects.</td>
</tr>
</tbody>
</table>

One important aspect of Action Diagrams is their semantic power to describe action logic (see Table 9-4). It is possible to describe sequential ordering of activities (i.e. the flow aspect), alternative activities (decision points), conjunctive activities (combinations), triggering (initiation) of activities (by time or communication), interruption of activities (by time or communication), contingent activities (i.e. activities occurring only sometimes), condition for activities, and parallel activities.

Table 9-4: Basic notation to describe action logic in Action Diagrams.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Ordered sequence" /></td>
<td>Ordered sequence of activities with action object as result and prerequisite respectively.</td>
</tr>
<tr>
<td><img src="image" alt="Suppressed action object" /></td>
<td>Suppressed action object. To be used only if the meaning is clear from the context, to increase readability.</td>
</tr>
<tr>
<td><img src="image" alt="Ordered sequence" /></td>
<td>Ordered sequence of activities with no intermediate action objects.</td>
</tr>
<tr>
<td><img src="image" alt="Alternative" /></td>
<td>Alternative.</td>
</tr>
</tbody>
</table>
Versatile information and business analysis

Conjunction.

Condition for action or action object. Often used in combination with alternatives and combinations.

Occasional action or action object (‘POSS’ is an abbreviation of ‘possibly’)

Activity triggered by communication.

Activity triggered by time.

Activity interrupted by communication.

Activity interrupted by time.

Parallel activities.

A contextual descriptive approach (Goldkuhl, 1992) is to be preferred when working with Action Diagrams. Each Action Diagram describes a business context within a business process. Different Action Diagrams are related to each other through descriptive connectors, i.e. through links to other Action Diagrams (see Table 9-5).

Table 9-5: Connectors used in Action Diagrams.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Sequential connector" /></td>
<td>Sequential connector to/from another Action Diagram. The Reference Code of the connected diagram is shown in the circle.</td>
</tr>
<tr>
<td><img src="image" alt="Hierarchical connector" /></td>
<td>Hierarchical connector to another Action Diagram that shows a decomposition of the connected activity.</td>
</tr>
</tbody>
</table>

To enable cross-referencing, all documents used in VIBA are given a unique Reference Code. This code is, for example, used to connect Action Diagrams this way. The demarcations of each Action Diagram (i.e. business context) are arbitrary; thus, an analyst is free to choose appropriate boundaries for each described context.

Action Diagrams make it possible to model and design the business and its information systems as an integrated whole. The actions of different performers – human actors and/or information systems (internal and/or external to the organization) – are described as a whole. Action Diagrams can thus be used to identify and delimit IS action. Such actions are described as integral parts of the business process. To say that IS actions are derived from the business process design is one way to put it. That business process design includes design of IS actions is equally correct. This counts for both interactive IS action (performed together with a user) and automatic IS action (performed by the computerized IS itself).

The Action Diagrams also show activities performed as a consequence of IS action. These IS-related actions form *use-situations*, which can be classified as (1) inter-
active, (2) automatic, or (3) consequential (see Chapter 4), and are used as a basis for delineating use-situation proposals (see Section 9.6.1).

9.5.3 Analyse Performers

Performers, communicators, interpreters, and the roles they play are identified during the activity Analyse Activity Structure (see Section 9.5.2). The purpose of analysing performers is to explicitly define these, and to investigate possible relationships between them. In Action Diagrams, the performers of activities are either information systems or roles played by the different business actors.

Important relationships to discover between roles are IS-A (i.e. inheritance in object-oriented terminology). For example a buyer might be able to act as receiver of goods but not the other way around. Hence ‘Buyer’ can be regarded as a special case of ‘Receiver’, that is, a ‘Buyer’ IS-A ‘Receiver’. These relationships are documented in Role Diagrams and the roles are defined in Role Definitions. These two documentation forms are similar to the Class Diagrams and the Class Definitions used during Interaction Modelling, which both follow the UML.

Actually, following UML, a role might be regarded as a special kind of class (at the meta-level); in UML’s terminology ‘role’ is referred to as ‘actor’. When it comes to design, identified and structured roles can be used as a source for maintaining access privileges to different parts of the constructed system(s).

It is also important to investigate the cardinality of different roles, and, if possible, enumerate the actual individuals cast to play each role.

9.6 Action Modelling

The aim of Action Modelling is to gain understanding of how the system is going to be used from an organizational perspective. It is important to understand that the actual use of the system in interactive use-situations is a topic of Interaction Modelling (see Section 9.7). One can say that Action Modelling maintains an abstract view of system usage while Interaction Modelling goes more into details.

Action Modelling consists of three activities, which are the responsibility of a Systems Analyst: Analyse Use-Situations, Identify Actions and Messages and Identify Documents.

9.6.1 Analyse Use-Situations

The aim of the activity Analyse Use-Situations is to understand in what business tasks the system is going to be used. This concerns all three types of use-situations: interactive and consequential as well as automatic. Use-situations are to be found in the Action Diagrams as adjacent activities and can analytically be classified into the mentioned three types.

The analysis of use-situations is structured into two main steps: Identify Use-Situations and Structure Use-Situations.

9.6.1.1 Identify Use-Situations

The purpose of identifying use-situations is to group together activities from the Action Diagrams into Interactive use-Situation Proposals (ISPs), Automatic use-Situation Proposals (ASPs), and Consequential use-Situation Proposals (CSPs). That is, to group

46 Actually it should be thought of as a role, played by an information system.
activities into chunks that are manageable from an analysis and design perspective. Identified use-situations are documented as a list of tables forming a Use-Situation List. The Use-Situation List serves two purposes. First of all, it shows explicitly what activities in the Action Diagrams that are aggregated to form the use-situations. In addition, it is a cross-reference showing which documents and messages that are relevant for the different use-situations.

In Section 4.5.1 ‘interactive use-situation’ was defined to be a primitive sequence of actions consisting of all interactive actions, and intermediate consequential, automatic and manual actions, that are performed adjacent in time by the same performers. This definition also counts for ISPs. Based on the definition, ISPs can be found by selecting arbitrary interactive activities in the Action Diagrams and traversing the flow of activities, in both directions (preceding and succeeding), until all adjacent activities, in each direction, not covered by the definition has been found.

In Section 4.5.1 ‘consequential use-situation’ was defined as all consequential actions, and intermediate automatic and manual actions, that are performed adjacent in time by the same performers, which are not part of any interactive use-situation. After identifying all ISPs, CSPs can be found by selecting arbitrary consequential activities in the Action Diagrams (that is, activities with no encapsulated information as output) and traversing the flow of activities, in both directions (preceding and succeeding), until all adjacent activities, in each direction, not covered by the definition has been found.

Finally, the ‘automatic use-situation’ was defined in Section 4.5.1 as all automatic actions that are performed adjacent in time by the same information system, that are not part of any interactive or consequential use-situation. ASPs can be found by finally selecting arbitrary remaining automatic activities (no human performers) in the Action Diagrams and traversing the flow of activities, in both directions (preceding and succeeding), until all adjacent activities, in each direction, not covered by the definition has been found.

The identified use-situation proposals are given consecutive identifying numbers and are documented in a Use-Situation List consisting of one table (similar to Table 9-6) per use-situation proposal (ISP, CSP, ASP) stating:

Â A descriptive name of the use-situation.
Â The messages needed to perform the use-situation (input messages) and the messages that are results of the use-situation (output messages).
Â The documents involved (used and/or produced) during the use-situations.
Â A set of references to the activities in the Action Diagrams in which the use-situation is described/used.
Â The performers involved in the use-situation.

If a use-situation corresponds to a single activity in the Action Diagrams, the name of that activity is normally used also for the use-situation. In other cases (i.e. more than one activity), it is important to find a name that reflects the actions performed. In both cases the Use-Situation List should contain references to the Action Diagrams and specific activities within them.
Table 9-6: Example, specification of an ISP.

<table>
<thead>
<tr>
<th>ISP 1: Feasibility Check (with respect to time)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input messages</strong></td>
</tr>
<tr>
<td>MD1: Preliminary Plan</td>
</tr>
<tr>
<td>MD7: Acknowledged customer orders</td>
</tr>
<tr>
<td>MD9: Material structures</td>
</tr>
<tr>
<td>Performers: Planner s MA</td>
</tr>
</tbody>
</table>

The performers involved are to be found in the Action Diagrams. Action Diagrams can also be used to find candidate input messages and output messages by looking at the information objects that are input to and output from the activities that constitute the use-situation at hand. Note that this might not yield a correct result since some messages found might need to be split in two or more smaller messages or several messages might need to be aggregated into one larger. Such needs will be identified during the activity Identify Actions and Messages later (see Section 9.6.2).

When viewing a use-situation proposal as a whole, there might be messages that do not seem to be outputs but are used ‘internally’. Such messages can be treated as both outputs and inputs since they are outputs of some activity, or activities, within the use-situation and inputs to some other(s).

The information about input and output messages contained in the Use-Situation List will be used subsequently during the activity Identify Actions and Messages (see Section 9.6.2).

In addition to messages, documents used to carry messages within each use-situation are also documented. These documents might be hard to identify initially and may need to be added as the activity Identify Documents is performed (see Section 9.6.3) and during the workflow Interaction Modelling (see Section 9.7).

The ISP List is an important document since it constitutes a cross-reference between the Action Diagrams and the more detailed documentation produced during Interaction Modelling.

If the current development effort embraces several information systems it is a good idea to give ISPs (and ASPs and CSPs) numbers such that the Use-Situation List is grouped by IS. This is however not always possible due to the fact that a single use-situation might involve several information systems.

When the ISPs and CSPs are identified and documented, it is time to check the Action Diagrams for automatic activities that are not included in any ISP or CSP. These will then constitute Automatic use-Situation Proposals (ASPs).

9.6.1.2 Structure Use-Situations

Sometimes there is a need to analyse how different use-situations are related to each other. This is done primarily to show the context of each use-situation and how use-situations and roles fit together. Such an analysis is documented in Use-Situation Diagrams, which show use-situation proposals as ellipses together with the actors involved
in each use-situation. The notation corresponds to the UML notation for Use case analysis (Booch et al., 1999). One use-situation might ‘use’ another use-situation. This means that interactions performed in many use-situations, for example logging on to a system, can be treated as a separate use-situation that other use-situations use. An inheritance relation between two use-situations shows the ‘use’ relationship between the two.

It is not necessary, even though possible, to show the relations between the different roles (Buyer, Supplier, et cetera) in a Use-Situation Diagram. As explained above, these are shown in Role Diagrams. Obviously, The Use-Situation Diagrams are extensions of the Role Diagrams showing not only the roles but also their relations to the different use-situations. One can think of these two diagrams as different views towards the same underlying model.

Sometimes it is most convenient to have only one of these two types of diagrams. At other times it is convenient to show the relations between roles separately. This is, for example, the case when only a few Use-Situation Diagrams are required due to few relations between use-situations.

9.6.2 Identify Actions and Messages

When analysis has proceeded so far that the Action Diagrams seem reasonably stable it is time to take a closer look at the messages being communicated in the business process. Messages (or ae-messages to be correct) can usually be identified in the Action Diagrams as information action objects flowing between activities. Information action objects that correspond to ae-messages should be identified and explicitly defined in Message Definitions. The most straightforward way to do this is by identifying all output messages in the Use-Situation List and create a Message Definition for each.

Message Definitions are important documents since much of the work during Interaction Modelling centres on them. A Message Definition states what e-action the ae-message is a result of, the character of that e-action (i.e. if it is interactive, automatic, et cetera), how the e-action depends on the action memory (i.e. independent, uses or updates), the communicator of the message, the performer of the corresponding e-action, and intended interpreters of the message. It is possible, and desirable, to identify and document also the communication function and corresponding communication effect.

To make the messages’ use contexts explicit, it is a good idea to indicate what activities the message is input to and output to by augmenting references to Action Diagrams and specific activities within those.

For messages created during interactive use-situations, the message definition also contains a reference to the document that creates the type of ae-message that is being defined. That is, a reference to the interactive screen document used to interactively perform the corresponding e-action.

Furthermore, a Message Definition states the propositional content of the message. This can be done in different ways, and typically the amount of detail increases as the analysis proceeds. At first, there might be textual references to concepts and properties. However, during Interaction Modelling when the propositional content is analysed in terms of classes, references can be made more specific. A label and, for example, an expression in Object Navigation Notation (ONN), as described by Blaha & Premerlani (1998), can be used to do this. Using ONN, and explicitly traversing the classes of the Class Diagram for the message, results in a verification of the information content. This is however considered being a part of the activity Analyse Concepts during Interaction
Modelling (see Section 9.7.2). Note that this means that the propositional content of an ae-message can be thought of as one view of a conceptual model of all the concepts used within the business. A Message Definition should usually also contain a brief description of the message it defines and how that message is used within the business.

One way to formulate the content of a Message Definition is to ask a series of questions for each action with corresponding message. These questions are shown in Table 9-7. The questions will need to be asked once for each identified interpreter (i.e. the ‘is directed towards’ row in the table).

Table 9-7: Questions to ask (and possible answers) during the activity Identify Actions and Messages.

<table>
<thead>
<tr>
<th>The action</th>
<th>Purchase order</th>
</tr>
</thead>
<tbody>
<tr>
<td>which on behalf of</td>
<td>the Paper Mill – Purchase dept.</td>
</tr>
<tr>
<td>is performed by</td>
<td>a Buyer</td>
</tr>
<tr>
<td>during the activity</td>
<td>PurInc [2]</td>
</tr>
<tr>
<td>is directed towards</td>
<td>a Supplier</td>
</tr>
<tr>
<td>with the aim of</td>
<td>ordering raw material</td>
</tr>
<tr>
<td>which hopefully will lead to</td>
<td>the delivery of raw material</td>
</tr>
</tbody>
</table>

In addition to these questions, we must of course ask ‘what are we talking about’? That is, what is the propositional content of the ae-message?

9.6.3 Identify Documents

As discussed in Chapter 4, ae-messages are created, carried and visualized by documents. At this point it is not possible to specify all documents that are going to be used in the IS; especially not interactive screen documents (ISDs), which are solely treated during Interaction Modelling. Some documents might however be identified during Action Modelling – typically paper documents, for example reports and other ‘output documents’. Documents are explicitly defined in Document Definitions.

A Document Definition for a paper document states which messages are being carried by the document, who the creator of the document is, in which ISP(s) it is being used, and a reference to a prototype of the document. Furthermore, it contains a brief description of the document and its intended use.

Note that not all information in Message Definitions and Document Definitions can be stated already during Action Modelling (for example references to classes in the content-part of Message Definitions and references to possibly not yet created Document Prototypes). Instead these documents are carried on to Interaction Modelling and continuously refined.

47 The creator of a paper document is either a performer (a role) using an interactive screen document (both shall be stated) or an automatic use-situation. If the creator of a paper document involves an ISD the creator information probably has to be left blank for now.
9.7 Interaction Modelling

The aim of Interaction Modelling is to get an accurate description of how users are going to interact with the system and to define concepts used to interact, and their relations to each other. The basis for Interaction Modelling is the proposed use-situations and the messages handled within them (as documented in the Action Set).

Interaction Modelling consists of the activity Analyse Interaction, which is the responsibility of a User Interface Designer, and the activity Analyse Concepts, which is the responsibility of a Systems analyst.

9.7.1 Analyse Interaction

To analyse interaction means to describe how the actual interaction between users and the system is going to be realized. Interaction most obviously takes place within interactive use-situations (ISPs). However, consequential use-situations (CSPs) usually contain interaction and must be considered during analysis of interaction.

Analyse Interaction consists of three main parts: Identify e-Interactions, Design Interactive Documents and Structure Interaction. It is important to note that these three activities are usually performed in a highly integrated manner. It is, for example, very hard to identify e-interactions without taking their representation in interactive documents into consideration.

9.7.1.1 Identify e-Interactions

To describe the e-interactions needed to formulate and send the messages of any given use-situation, VIBA suggests Interaction Tables (I-Tables), which build explicitly on the EIAL (see section 4.5.2). An I-Table is a table with three rows and three columns. The left-hand column is used for the user actions, the middle column for the state of the current interactive document(s) and the right-hand column is used for the IS actions. One I-Table is used for each e-interaction (EIAL).

I-Tables are grouped into e-Interaction Lists. One e-Interaction List is produced for each use-situation proposal. An e-Interaction List consists of three parts. First there is a list of initial IS actions that are to be performed upon entry to the use-situation. Second, there is a listing of the e-interactions to be performed in the use-situation. Finally, there is one I-Table for each e-interaction that specifies the interaction in detail (see Table 9-8).

From the I-Table describing e-interaction 3.3 in Table 9-8 we see how $s_0$, cell (1,2), specifies that the ‘Purchase order’ document, with information about the ID of the current order, the order date and the buyer’s reference person, is to be visible in order to perform e-interaction 3.3, i.e. a precondition for the e-interaction. Cell (2,1) specifies what e-interaction the I-Table is concerned with, i.e. associating a supplier to a purchase order. It is sometimes unnecessary to show the initial state $s_0$ within an I-Table, and hence the first row can be omitted.

Cell (2,2) specifies the state of the e-interaction after the user action and the specific GUI components used to perform the e-interaction. In this case a combo-box labelled ‘Sup.ID’. The supplier is now associated to the purchase order (from the user’s point of view). Column 3 shows the actions that are to be performed by the IS in response to the user’s action. Cell (3,2) shows the response from the IS, i.e. the purchase order document with supplier information shown. Finally, cell (3,1) shows the interpretation act to be performed by the user. In this case the interpretation should yield either that the correct supplier has been associated with the purchase order or that another
supplier being chosen, which, in the latter case, leads to the re-performance of the same e-interaction.

Table 9-8: Example, I-Table from an e-Interaction List that specifies e-interaction 3.3.

<table>
<thead>
<tr>
<th>3.3</th>
<th>User Action</th>
<th>Document</th>
<th>IS Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>S₀</td>
<td>Purchase order</td>
<td>Order ID, Date, Our Reference</td>
<td></td>
</tr>
<tr>
<td>S₁</td>
<td>Associate supplier with purchase order</td>
<td>Purchase order</td>
<td>Retrieve information about selected supplier and show in all supplier-related fields</td>
</tr>
<tr>
<td>S₂</td>
<td>Interpretation</td>
<td>Purchase order</td>
<td>All supplier information visible</td>
</tr>
<tr>
<td></td>
<td>1. Correct supplier associated</td>
<td>“Supplier ID” (cbSupID)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2. Wrong supplier associated (6.3.3)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In most cases the number of IS actions that correspond to a user action are one or perhaps two in strict sequence. Sometimes there might however be several IS actions involved and thus a need to describe how these relate to each other. In such cases, for example, a regular expression specifying the sequence can be augmented to column 3. Sometimes it is preferable to instead use, for example, an Object Navigation Notation (ONN) expression (Blaha and Premerlani, 1998) that dictates what information to show, and thereby omitting to enforce any sequence restrictions. If sequence restrictions are important from a business perspective they shall be explicitly shown.

The level of detail in the documents showed in I-Tables depends on how far the analysis has proceeded. At first, the documents are referred to by textual references (as in Table 9-8). When the analysis proceeds, the layouts of the documents get more and more detailed. This evolution can (and should) be shown in the I-Tables by inserting thumbnails of documents or document parts in column 2.

The language used within I-Tables is natural language such as English or Swedish. VIBA do however suggest some conventions that semi-formalize the language use, as described in Table 9-9.

Table 9-9: Notational conventions used in I-Tables.

<table>
<thead>
<tr>
<th>Convention</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraints regarding the performance of user actions are written surrounded by curly brackets in cell (2,1).</td>
<td>{By drag ‘n drop 2→1} means that the user action is performed by use of drag ‘n drop from document 2 to document 1 as numbered in cell (1,2).</td>
</tr>
<tr>
<td>Constraints regarding the performance of IS actions are written surrounded by curly brackets in column 3.</td>
<td>[RT ≤ 3s] means a response time equal to or less than 3 seconds.</td>
</tr>
<tr>
<td>Clickable items are GUI components used to execute the user action. These are surrounded by quotation marks in cell (2,2).</td>
<td>“New order” is a clickable item labelled ‘New order’ used to create a new order.</td>
</tr>
</tbody>
</table>
References to names given to GUI components in a prototyping tool might be shown after their ‘visible’ caption as shown in the GUI.

| “Order ID” (cbPurOrderID) means that the clickable item ‘Order ID’ is named cbPurOrderID in the prototyping tool. This also indicates that it is a combo-box due to the ‘cb’-prefix (using Hungarian notation). |
| To reference a sequence of IS actions the function sys can be used. |
| Sys(3.1) means that the IS actions stipulated in I-Table 3.1 shall be performed also here. |

Note that there is no source (i.e. no previously created deliverables) that can be used to analytically find e-interactions. Instead e-interactions are typically found while prototypes are being constructed and tested during the activity Design Interactive Documents.

9.7.1.2 Design Interactive Documents

The purpose of designing interactive documents is to operationalize the visualization of actability into the system under construction. That is, to make sure that information gathered about the various actions in which the system is to be used is really made visible in the user interface. One way to do this is to utilize the Message Definitions to see the aim of the messages created by use of an interactive screen document: the intended interpreters, communication functions, and communication effects. All these aspects should be reflected in the screen document, by, for example, carefully selected labels and captions.

Various interactive document design aids, such as Nielsen’s (1993) heuristics and Shneiderman’s (1998) golden rules, as well as the actability principles presented in Chapter 12 can be used as guidelines during this activity. Much of the required information can be derived from the Message Definitions, the Use-Situation List and the Action Diagrams. However, during this activity the most natural source of information is the interacting end users – the human performers.

When designing documents – interactive and others – during Interaction Modelling, the focus is on the particular use-situation under investigation. Some documents, however, are used in many use-situations and so there is a need to carefully verify that no inconsistencies arise. One way to assure this is to create a Document Definition for each document. The Document Definition serves a double purpose. First of all, it is a description of the particular (type of) document. Additionally it serves as a cross-reference between the Message Definitions and the documents used to carry and create the messages.

A Document Definition consists of six sections. First, there is a verbal description of the document briefly describing its intended use. Subsequently, there are references to message(s) carried by the document (that is, input messages in the ISPs in which the document is used), message(s) created by the document (that is, output messages from the ISPs in which the document is used), document(s) created by the document, which are usually paper documents, the ISPs in which the document is used, and to a prototype of the document. Such a Document Prototype is a visualization of a document, usually created during interaction analysis in parallel with I-Tables and sequence restrictions. Note that prototypes of paper documents might be created during Action Modelling in conjunction with their Document Definitions.

When referring to a document, in Document Definitions and throughout the documentation whenever a document is referenced, the name of the document is prefixed by a stereotype stating what type of document we are dealing with, for example: a paper document.
document, or an interactive screen document. The tentative classification we have used so far is showed in Table 9-10.

Table 9-10: Stereotypes used for different types of documents.

<table>
<thead>
<tr>
<th>Stereotype</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>«PD»</td>
<td>Paper Document</td>
</tr>
<tr>
<td>«ISD»</td>
<td>Interactive Screen Document</td>
</tr>
<tr>
<td>«SSD»</td>
<td>Static Screen Document (non interactive)</td>
</tr>
</tbody>
</table>

9.7.1.3 Structure Interaction

Each I-Table describes one e-interaction so there is a need to describe how these are related to each other. As described in Chapter 4, the action potential of any given interactive screen document might vary over time. Since action sequence restrictions constitute the state of the document, and thereby restrict action potential, we use the formalism of Statecharts (Harel, 1987) to model them.

A related issue is the modelling of navigation paths between documents, which could be done with I-Tables but such an attempt would not be feasible due to the many possible navigation paths. Instead, this is also done with Statecharts where each high-level state represents a document and each navigational action is represented by a transition. The resulting state model can be viewed as consisting of documents as high-level states where the action sequence restrictions of each document (see Figure 9-3) correspond to sub-states of those.

This approach to model action sequence restrictions is an extended version of Horrocks’ (1999) Statechart approach to user interface construction. With Horrocks’ approach there is a clear path from interface design to implementation in, preferably, event driven rapid development tools (for example, Visual Basic and Borland Delphi). However, his approach lacks a clear connection to business actions, which is provided by the use of e-interactions as a source for transitions rather than Horrocks’ vaguely defined notion of event.

Since the suggested approach is an extension of Horrocks’ (1999) approach, the techniques and help-documentation he suggests, and thoroughly discusses, might help when performing interaction analysis.

When structuring interaction, there are additional considerations regarding the design of interactive screen documents. We must, for example take into account how the information in documents depends on what document was shown before the one at hand. We must also consider in what order things are allowed to occur – in what order actions and interactions can be performed. It would, for example, make sense not to allow for an order to be registered before a customer has been associated with it. Sequence restrictions should generally be avoided, but are sometimes necessary.
9.7.1.4 Additional Considerations

The activity Design Interactive Documents concerns interactive and (interactive) consequential use-situations and hence interactive documents have been in focus so far. The documents used in consequential and automatic use-situations are often based on ae-messages from interactive use-situations and they are therefore normally treated during interaction analysis. However, there is a need to ‘walk through’ the business model and make sure that all the documents have been taken into account. There might also be a need to consider all ae-messages and documents as a whole. The partially fragmented document model is then consolidated into a system global level.

9.7.2 Analyse Concepts

In order to design interaction and interactive documents as well as automatic actions, there is a need to consider the propositional content of messages. This is preferably done during a conceptual analysis based on identified ae-messages.

Conceptual analysis and analysis of interaction must be performed in integration. During Design Interactive Documents the professional language used by the actors is captured and used in interactive documents. This language is now being analysed using ‘traditional’ conceptual modelling in terms of concepts (classes) and properties, as done in object-oriented analysis (Graham, 1998; Booch, 1994; Blaha and Premerlani,
See also Chapter 6 for a discussion about an action-oriented approach to conceptual modelling, which should serve as background for this activity.

9.7.2.1 Define Concepts

Each discovered class (concept) is defined in a Class Definition, which, following UML, consists of a brief description, the cardinality of the concept, i.e. how many instances are allowed and/or supposed, the class’ position in a possible inheritance hierarchy by stating super-classes, attributes; and associations. Note that the same class might be used in different ae-messages. The Class Definition, however, is a ‘central’ description of the properties of a class that holds for all ae-messages. Hence, not all attributes and associations defined in the Class Definition are shown in all Class Diagrams. Note also that the e-actions usually constitute classes, as described in Chapter 6.

9.7.2.2 Structure Concepts

To document conceptual relationships between concepts VIBA makes use of UML Class Diagrams, where one class diagram is produced for each ae-message. One can, following Graham (1998), argue that UML is not fully suitable for this kind of modelling due to its bias towards relational concepts, such as allowing for constraints to be attached to relationships (which hence are not encapsulated by any class and so break the principle of encapsulation). However, UML can be used, with caution, and it is arguably the best choice available due to its widespread use and the fact that it is a de facto standard in many software organizations. Furthermore, VIBA is intended to create specifications suitable for object-oriented development as well as for traditional relation-oriented development.

9.7.2.3 Verify Conceptual Consistency

The same concept might very well be used in several messages. Therefore it is important to verify that no inconsistencies arise. Verify Conceptual Consistency thus means to go through the Class Definitions and the Message Definitions in order to verify that the definition of every single concept holds for all messages. Note that through the use of a method tool (CASE) it is possible to do conceptual analysis based on ‘parts’ of an object model, for example messages, and still maintain global consistency. This is important in order to verify that the same class (concept) is not used with different definitions in different contexts.

9.7.2.4 Analyse Behaviour

Following the UML, it is possible to also attach a state machine to a class in order to describe its dynamic behaviour. ‘Dynamic behaviour’ refers to the fact that classes might show different properties during different ‘phases’ or ‘states’ of their existence. This is also a way of describing what e-actions create, modify, and destroy, instances of a particular class. These are important properties when designing the action memory (during, for example, database design). Note that this is also a way of pragmatizing dynamic conceptual modelling. That is, there is an explicit connection between business actions and the dynamic model of the system, something that is often unclear in contemporary approaches to conceptual modelling, such as those proposed by Blaha and Premerlani (1998) and Snoeck and Dedene (1998). Analysing behaviour is not always necessary or considered affordable (Blaha and Premerlani, 1998). If it is done, a reference to the Class Statechart showing the state machine should be included in the Class Definition.
9.8 Conceptual Overview of VIBA

This section provides a concluding summary of VIBA by explicitly relating the different deliverables to each other via the concepts they document as an aid to understanding how the VIBA documentation forms a view of the underlying conceptual model of the system that is under construction.

Different types of documents, and the software system itself, represent the different deliverables produced during systems development. A document should be thought of as a view of an underlying model of the system and its surrounding and rationale. The different elements shown in the documents (for example, classes, roles, documents) are all visual representations of the concepts used to talk about the system – so-called modelling primitives. In this section we look at how the different documents suggested by VIBA relate to their underlying concepts. From the understanding of these relations it is possible to deduce how the different documents relate. In the description, concepts (modelling primitives) are shown as tiny clouds.

In the following description two basic relationships between concepts and document types are shown as *defines* and *relates*. First, there are several deliverables (documents) that serve to define a particular instance of a concept. As an example, consider the Class Definition, which defines the static characteristics of a certain class. Second there are documents that relate different instances of concepts to each other. An example is the Class Diagram, which shows how several classes are related to each other to form the informational (or propositional) content of a message. There is an additional type of relationship worth mentioning, even though it is only used ones in the following description. It is the relationship described as *visualizes* between the concept Document and the deliverable Document Prototype that shows the visual representation of the document. This can be thought of as a special form of definition since, in some sense, the prototype defines the visual representation of the document.

9.8.1 The Prerequisites Set

The aim of the Prerequisite Set is to document the prerequisites and high-level requirements, and to define the scope of the new system. Figure 9-4 shows the types of documents it contains and their relationships to the concepts focused on during modelling.

The focus is on the business and the problems it is experiencing, the goals that solve those problems and the vision of a new system that realizes those goals. Attention is also turned towards other, supplementary, prerequisites imposed by the way business is to be done and the vision. Other supplementary requirements can be traced from the development situation as such, for example available time for development. Another part of the Prerequisite Set is the Glossary, documenting preliminary definitions of central concepts in the business as well as project-specific concepts not related to the communication that the system is going to be a part of.
9.8.2 The Activity Set

The aim of the Activity Set is to document the action structures of the business defined in the Prerequisite Set. Figure 9-5 shows the types of documents it contains and their relationships to the concepts focused on during modelling.

The focus is on the activities performed and on the performers of those activities. As an integral part of the action structure the various action objects (ae-messages/documents and material objects) are considered. Action objects are both results of activities and prerequisites (or at least input) to activities.
9.8.3 The Action Set

The aim of the Action Set is to document how the new system is going to be used from an abstract point-of-view. It is concerned with the use-situations that aggregate activities from the Action Diagrams into manageable analysis pieces. It is also concerned with the e-actions performed during activities, and hence during use-situations. Furthermore, the communicative results of e-actions, ae-messages, are explicitly documented. Figure 9-6 shows the types of documents contained in the Action Set and their relationships to the concepts focused on during modelling.

![Diagram of Action Set concepts](image)

Figure 9-6: Action Set deliverables and primary concepts.

It is important to understand that the Action Set applies a broader perspective to the action structure than the Activity Set in that it focuses activity aggregates. At the same time it applies a narrower perspective in that it focuses on e-actions and ae-messages that build up activities.

9.8.4 The Interaction Set

The purpose of the Interaction Set is to document in detail how the users are going to interact with the system. The focus is on documents, primarily interactive documents. During interactive use-situations, e-interactions are performed in order to formulate and send ae-messages – i.e. to execute e-actions – through the system. During consequential use-situations, e-interactions are performed in order to get sufficient background information for performing actions outside of the system. Both types of use-situation are thus built up of e-interactions, but they serve partly different purposes in each. Figure 9-7 shows the types of documents contained in the Action Set and their relationships to the concepts focused on during modelling.

Note that the relationship between the Document Prototype and the Document concept is named ‘visualizes’. This can be thought of as a special kind of definition, as discussed above.
9.8.5 The Conceptual Set

The purpose of the Conceptual Set is to document the concepts used in the various ae-messages. The concepts used in ae-messages represent its *propositional content*. Concepts are defined in Class Definitions. Concepts are usually related to each other in different ways to form a meaningful whole. Such relationships are documented in Class Diagrams – usually one per ae-message and possibly one consolidated system global diagram\(^{48}\). Concepts might inhabit behaviour – change their state during the course of actions in the business. Such dynamics can be defined in Class Statecharts.

Figure 9-8 shows the types of documents contained in the Conceptual Set and their relationships to the concepts focused on during modelling.

---

\(^{48}\) Class Diagrams can be created as views towards the conceptual model at arbitrary levels of granularity.
Chapter 10

Actability Design: Evolution and Empirical Lessons Learned

This chapter presents lessons learned from working with the VIBA method, which was presented in Chapter 9. As discussed in Chapter 8, VIBA has evolved during the work and exists in five different externalized versions. To position the empirical findings within this evolutionary process, the chapter starts with a discussion about the evolution of VIBA and its relation to the underlying concept of actability.

10.1 Evolution of VIBA

In this section, the main differences between the five different major externalized versions of VIBA are explained – from the original VIBA’93 to VIBA’02, which is the version presented in Chapter 9. Additionally, some modifications made between the intermediate VIBA versions are described as an aid to understand the discussion about empirical results subsequently.

10.1.1 From VIBA’93 to VIBA’02

VIBA’93 consists of four main focal areas (Goldkuhl, 1993) as shown in Figure 10-1 (author’s translation from Swedish): Business Analysis, Message Analysis, Processing Analysis, and Effect Analysis.

![Figure 10-1: Main focal areas of VIBA’93 (Goldkuhl, 1993).]

10.1.1.1 Business Analysis

The Business Analysis of VIBA’93 corresponds roughly to the Process Modelling in VIBA’02. The aims of the two are the same, but the activities performed are grouped somewhat differently. Business Analysis consists of four primary activities (author’s translation from Swedish):

Á Goals Inventory,

[49] These versions were first introduced in Table 8-4, p. 144.

Goals Inventory corresponds to the Process Modelling activity Analyse Prerequisites. The name has been changed to reflect the fact that an inventory of goals is actually performed as one element amongst others during this activity.

Business structuring corresponds to the Process Modelling activity Analyse Activity Structure. The name has been changed to emphasize that it is the structure of activities (action aggregates) that is analysed.

During Computer systems overview, the different computerized information systems found during Business structuring are described in terms of different functions (messages sent and received) and involved actors.

During Work task analysis, the different tasks to be carried out are described as well as the actors performing the tasks.

These last two activities have been regrouped in VIBA’02 to better match the concept of use-situation as suggested by actability (see Chapter 4). In essence, a work task corresponds to a use-situation and hence the activity Analyse Use-Situations corresponds to a large part of Work task analysis. However, we have chosen to treat all performers (actors and information systems) together during Analyse Performers in VIBA’02, instead of treating information systems during Computer systems overview and actors during Work task analysis as done in VIBA’93. This accords better with the notion of performer as suggested by actability (see Chapter 4).

10.1.1.2 Message Analysis and Processing Analysis

In VIBA’93 Message Analysis consists of four activities (author’s translation from Swedish):

- Message Definition,
- Concept Analysis,
- Information Structuring, and
- Dimensioning.

In addition, Processing Analysis consists of four activities (author’s translation from Swedish):

- State Analysis,
- Processing Design,
- Dialogue Design, and
- Processing Rules Definition.

The aim of Message definition is to define the messages sent and used through information systems in the business. This is performed during the activity Identify Actions and Messages in VIBA’02. To treat this as part of the workflow Action Modelling seems more appropriate since messages are considered as results of actions (see Chapter 5) – actions that aggregate to activities.

Concept Analysis (i.e. defining the concepts used), Information Structuring, Dimensioning, and State Analysis, constitute together the activity Analyse Concepts in VIBA’02.

Processing Design, Dialogue Design, and Processing Rules Definition have been replaced by the activities Analyse Interaction and Identify Documents in VIBA’02. As
shown in chapter 7, these two activities together constitute a better match with the actability notion of formulation and sending of ae-messages, as discussed in Chapter 4; they are actually directly derived from that notion.

10.1.1.3 Effect Analysis

Effect Analysis consists of three activities (author’s translation from Swedish):

- Economic valuation,
- Work situation analysis, and
- Design valuation.

In VIBA’02 we have chosen not to treat this as a separate focal area with separate activities. Instead, these are aspects to consider during a continuous reflective evaluation. However, if time permits, it is possible to adopt the Effect Analysis of VIBA’93, and perform a comprehensive analysis of the effects of the proposed design. This might be specifically useful when a waterfall model is used during development, since there are no ways of adjusting the requirements during later phases in such cases. It lies outside the scope of this dissertation to go into the details of Effect Analysis but certainly, the presented Actability Evaluation method (see Part IV of the dissertation) can be used to do summative (post-implementation) as well as formative evaluation of a system developed by use of VIBA.

10.1.2 Process Modelling versus VIBA’93

As was pointed out above, the workflow Process Modelling bears considerable resemblance to the Business Analysis of VIBA’93. The activities are grouped differently and there have been several changes in the documentation forms, but several document types have also been retained. The document types that are unchanged (or at least very similar) are: Business Definitions, Problem Lists, Problem Diagrams, Goal Lists, Goal Diagrams, Supplementary Prerequisites and Action Diagrams. In VIBA’93 there is no notion of a systems vision and, consequently, no Vision Document. Instead this was considered part of the Business Definition. In VIBA’00, the vision was made explicit and the Vision Document as a high-level steering instrument was introduced, inspired by the similar notion in RUP (Kruchten, 1999) and the eXtreme Programming concept of metaphor (Beck, 2000). The Glossary of VIBA’02 was introduced as a way of capturing the business language informally throughout the development process.

In VIBA’93 there is no notion of ‘document’ in the way suggested by actability and VIBA’02. Consequently, neither are there any Document Definitions. Instead the media is described together with the definition of each particular message. This is done in so-called Message Summaries. The Role Diagrams and Role Definitions have no directly corresponding document types in VIBA’93. Instead there are IS Summaries and Unit Summaries.

An IS Summary describes what functionality a particular IS is supposed to have, what messages it uses and produces, and for each message its sender and receiver. In VIBA’02, this information is to be found in the Use-Situation List, grouped by use-situation instead of IS. Of course, the information can easily be extracted from the Use-Situation List and grouped by performer if needed.

A Unit Summary corresponds to a Role Definition for organizational units. In VIBA’02 Role Definitions are used for all performers (not just organizations), communicators, and interpreters. This is a consequence of the distinction between different
kinds of performers, and the separation between communicator and performer, in act-
ability. The relations between these are not clearly articulated in VIBA’93.

10.1.3 Action Modelling versus VIBA’93

As indicated above, the workflow Action Modelling corresponds partly to Message
Analysis in VIBA’93. Message Definition is now considered as part of the activity
Identify Actions and Messages and hence separated from Analyse Concepts of the
workflow Interaction Modelling.

In VIBA’93, a Work Task Description contains a description of a particular activ-
ity. In essence, this is a more verbal description of a particular use-situation, even
though the notion of use-situation is not present in VIBA’93. In VIBA’02, the use-
situation concept is explicitly utilized and use-situations are documented in Use-
Situation Lists and Use-Situation Diagrams.

In VIBA’93 there is no specific analysis of documents, instead so-called Layouts
(sketches of documents) are created during Message Analysis in conjunction with the
creation of Message Summaries. In VIBA’02, this is done during Identify Documents,
which yields Document Prototypes and Document Definitions of paper documents, and
Design Interactive Documents during workflow Interaction Modelling. In addition, the
creation of Message Definitions is separated from the allocation of messages to docu-
ments, in order to distinguish the abstract concept of the ae-message from its concrete
representation in documents.

10.1.4 Interaction Modelling versus VIBA’93

What is left of Message Analysis and Processing Analysis in VIBA’93 corresponds to
the workflow Interaction Modelling. One important difference between VIBA’93 and
VIBA’02 is that in VIBA’02 the notation for conceptual analysis is borrowed from
UML whilst VIBA’93 proposes a method-specific variant of traditional ent-
ity/relationship notation. One high-level design goal when re-designing VIBA was to
make use of existing notations and modelling formalisms. Therefore, it seems natural
to use UML for this purpose.

VIBA’93 does not explicitly promote the use of prototypes, even though this is
mentioned as a possible complementary approach. Instead human-IS interaction is
modelled with Interaction Diagrams in VIBA’93. Interaction Diagrams are, in essence,
fine grained Action Diagrams that describe the interaction in a procedural way. Such an
approach do not lend itself very well for development with event-driven rapid applica-
tion development tools, which VIBA’02 aims to support. A state-based approach is
more appropriate to model event-driven software (Horrocks, 1999), which is why the
Statechart modelling is proposed by VIBA’02.

In VIBA’93, IS actions (which were referred to as functions) are described in
Processing Diagrams and Function Descriptions. A Processing diagram is, like Interac-
tion Diagrams, a fine-grained Action Diagram describing processing logic proce-
durally. A Function Description describes how different business rules apply to each
function of the IS. As seen in the discussion about formulating and sending ae-
messages in Chapter 4, the aim was to make the connection between business context,
human-IS interaction, and IS functionality more explicit than it is in VIBA’93. There-
fore, I-Tables are used to describe IS functionality (IS action) integrated with the
description of formulation and sending of ae-messages. In VIBA’02 it is also possible to
describe IS-action by use of ONN expressions (see Section 9.7). Hence, one is not
restricted to procedural descriptions but can use declarative statements when appropriate.

10.1.5 VIBA’98 versus VIBA’02

The IS specification created using VIBA’98 during Application 2 (see Sections 8.3 to 8.6) consisted of the following documents:

- Action Diagrams,
- Document Definitions,
- ISP List (referred to as Use-Situation List in VIBA’02),
- I-Tables,
- Document Prototypes,
- Statecharts (Document Statesharts),
- Class Diagrams, and
- Class Definitions.

The Action Diagrams correspond to the ones in VIBA’02, while the Document Definitions have undergone important changes (changes that were effective already in VIBA’99).

When working with the document metaphor as a generic concept, including screen documents, we had, as we during Application 2 finally understood, confused the concept of the ae-message with that of the document. Under the false assumption that one document type carried one and only one ae-message type, we had put the descriptions of propositional content, action mode, \textit{et cetera}, in the document definitions. Hence, the concept of message had been somewhat underrated in VIBA’98, compared with the original VIBA’93. This, of course, also affected the externalization of actability. Even though the concept of the ae-message was included in actability, its relationships to documents were not made explicit.

As a consequence of the ‘re-introduction’ of the message concept, the ISP List came to look slightly different also. In VIBA’98 the ‘input message’ and ‘output message’ was missing. Hence, the ae-message concept as a bridge from Action Modelling to Interaction Modelling was not as emphasized, and thus not as utilized as an analytical tool, as proposed in VIBA’02.

The I-Tables and Document Prototypes look similar in VIBA’98 and VIBA’02. The Statecharts, however, have changed somewhat. Of course, the formalism as such has not changed, but the way in which we use it has. In VIBA’98 the ISP was in focus when modelling sequence restrictions; in fact ‘Sequence Restrictions’ was the phrase used for the Document Statecharts. We also used the distinction between \textit{user actions} and \textit{navigational actions} to treat these in two separate models, both based on Statecharts. However, influenced by Horrocks (1999), we have decided to treat them together in VIBA’02. The Class Definitions and Class Diagrams still follow UML and hence look the same. However, VIBA’98 suggested creating one Class Diagram per use-situation proposal, instead of one per ae-message, as in VIBA’02. The VIBA’98 approach seems valid from a ‘task’ or ‘use case’ perspective, but when following a language/action perspective, the VIBA’02 approach seems more intuitive. In many cases these two approaches yield the same result, such as when one (type of) ISP ‘produces’ one message type. However, if the same message type is used in several ISPs, the VIBA’02 approach has proved to reduce redundancy in the documentation.
10.1.6 VIBA’99 versus VIBA’02

The main difference between VIBA’99 and VIBA’02 consists of a restructuring of the activities of the method (in addition to the introduction of a few new deliverables, such as the Vision Document and the Glossary). VIBA’99 was structured into two main areas, referred to as Business Process Modelling (BPM) and Information Systems focussed Modelling (ISM). Figure 10-2 depicts the overall ‘high-level’ structure of VIBA’99.

When moving on to VIBA’00, this structure changed into that of VIBA’02 (even though the names of the workflows has been changed since VIBA’00). The reason for this restructuring was mainly for clarity. During Application 3, it was realized that students had difficulties in remembering the different activities that belonged to the different parts of VIBA. We therefore decided to divide the overall structure into three parts, each with less content, based on the assumption that it is easier to remember three times six things than two times nine. The restructuring also matched the distinction made in actability between the levels of activities, e-actions and e-interactions.

10.1.7 VIBA’00 versus VIBA’02

As indicated above, the differences between VIBA’00 and VIBA’02 are subtle, mainly consisting of refined terminology. The major difference is that VIBA’00 (as well as VIBA’98 and VIBA’99) was based on the view of the ae-message as described in Sec-
10.2 From Actability to VIBA

VIBA was described in Chapters 1 and 3 as an operationalization of actability. In this section the main relations between VIBA and actability are commented upon, in order to clarify how the operationalization has been made.

10.2.1 Social Action as the Starting Point

The most obvious connection between VIBA and actability is perhaps the central roles that e-actions and ae-messages play. During Process Modelling the action structure of the business processes analysed is documented in Action Diagrams. The Action Diagrams then serve as a basis for delineation of use-situations (which is another central actability concept). The dual character of information (propositional content and action mode) as proposed by speech act theory, and inherent in actability, is fully considered when activities and messages are analysed in VIBA.

Another central issue in actability is the distinction between performer and communicator. Communication action is intentional and it is hence important to discuss and define all possible performers of actions during systems development. Since information systems are regarded as performers within actability, it is important to distinguish between the person or IS who performs the action and the person who is actually doing something in relation to interpreters. This is recognized to be one of the most important parts of the activities termed Analyse Activity Structure and Analyse Performers, as discussed in Sections 9.5.2 and 9.5.3.

Another implication of the fact that action is considered intentional is the analysis of goals during Analyse Prerequisites. Identified goals can serve as an analytical tool to verify that the business’ goals are made operational by corresponding actions (cf. Ågerfalk and Åhlgren, 1999).

10.2.2 Speech Acts and Interaction Modelling

The baseline for Interaction Modelling and the activity Analyse Interaction is the actability division of speech acts into five different components (see Section 4.5.2, p. 73). According to actability, a use-situation (referred to as use-situation proposal during systems development, i.e. ISP, ASP and CSP) consists of the formulation and execution of one or many e-actions. Each performance of an e-action can be viewed as consisting of five different parts (following the discussions in chapter 4):

a) A formulation phase.

b) An execution.

c) A propositional content.

d) One or several Communication functions.

e) One or several Communication effects.

The first and second part (a and b), can and should be supported by the information system – this is what Analyse Interaction is about. Each interaction with an IS, performed as to formulate ae-messages and execute e-actions, follows a structure called the Elementary InterAction Loop (EIAL). The I-Table notation builds explicitly on that
structure. However, since the formulation act consists of a sequence of interactions, we need a way to describe sequence restrictions – this is done with Document Statecharts.

The third part (c), concerns the conceptual structure of the resulting ae-message – this is what Analyse Concepts is about. This also implies how the design of the action memory is affected by the different e-actions. During Analyse Concepts, concepts are explicitly defined in Class Definitions, and their relationships with each other are described in Class Diagrams. If some important properties of a class (concept) change over time, it is possible to describe the ‘life cycle’ of a class in a Class Statechart.

The fourth and fifth parts (d and e), form the point of departure for the analyses performed during the activity Analyse Actions and Messages.

10.3 Empirical VIBA Lessons Learned

As explained in Section 8.3, the empirical work has resulted in ideas and data, respectively. When interpreting the data, and drawing conclusions from it, it is thus necessary to be aware of that ideas have changed the design, to which data refers, both during and after data collection and interpretation. The data collected during the realization phase of Application 2 therefore were interpreted in the light of what the specification looked like when handed over to the consultants (see Section 8.3), referred to as VIBA’98. Similarly, data collected during Application 4 were interpreted in the light of VIBA’00. It is therefore important to be aware of the differences between what the specification looked like at these times and what is described in Chapter 9 (VIBA’02), as discussed in Sections 10.1.5 to 10.1.7. The remainder of this section presents the result of the analysis of recorded data as explained in Sections 8.3 to 8.6. From the data two major categories have been identified, process and product. These two have been used as a means of structuring the presentation of the results. The process category contains issues relating to the systems development process, and the product category contains issues relating to the specification (i.e. the documents (deliverables) produced during the process). The process category has been sub-categorized into issues regarding the overall development process model and those regarding specific modelling activities. The product category has, similarly, been sub-categorized into the different types of documents used in the VIBA’98 and VIBA’00 specifications, respectively. The section ends with some general concluding remarks. Throughout the text, the labels given to the different sources of data as described in Table 8-3, p. 142 (the What and Source columns), are used. Note that quotations are the author’s translation from Swedish.

10.3.1 Process

10.3.1.1 Development Process Model

VIBA proposes a development process model based on the RUP (Kruchten, 1999) which allows for flexible adoption. Analysts are supposed to tailor the development process model so that it suits the particular development effort at hand. For example, the model can be applied very iteratively with controlled iterations and incremental delivery. During both Applications 2 and 4 we were forced to rely partly on a waterfall model. That is, the development process model was applied without iterations between the VIBA workflows and the following realization activities. During requirements elicitation and analysis in Application 4 we (R1 and R3) worked iteratively with analytical models and prototypes. But since the specification was handed over to the IT consultants, the specification constituted a milestone and implied a waterfall. This
resulted in several criticisms from the IT consultants, the most important being that they felt that the specification was fixed and that they had difficulties knowing when and what to change. And changes were necessary due to both technical restrictions and to budget reasons. What should have been done was probably to overlap the project team doing analysis with that doing realization. As it turned out, the specification had seriously reduced the available design space, which led to frustration.

Similar problems were identified also in Application 5. In this case two researchers (R3 and R4) did most requirements elicitation and analysis, while another person (R5) implemented the functionality in the system (which was in the form of working prototypes with live data). During interviews I5, I6 and I7 all three participants suggested, similar to C1 and C2 (see above), that it would have been better had R5 participated in the earlier stages of the project also. As it turned out, changing requirements eventually led to re-analysis during which R5 participated. Both R4 and R5 considered this as positive.

Actually, problems of this sort were expected with the chosen approach (cf. Section 8.2), and led over to another observation made during Application 1, and utilized during the analysis phase of Application 2.

The observation concerns the integrated approach to business modelling and human-IS interaction modelling proposed by VIBA. What we observed was that assumptions about the business process activity structure, manifested in Action Diagrams, came to change during the Analyse Interaction workflow. When digging deeper into the action structures and letting users work with prototypes, the business model changed. One conclusion drawn from this is that an integrated approach, such as the one proposed by VIBA, is necessary to develop an accurate business model.

### 10.3.1.2 Process Specifics

One important lesson to learn from Application 1 is related to the prototyping approach. The heart of the IS was production planning, including optimization of production schedules. To be able to build a prototype that showed functionality related to this, we had to design the optimizing algorithm. Unfortunately, that was easier said than done. Since the algorithm, in turn, relied on the underlying data structure, we had, in practice, to come up with the complete system to be able to visualize that part of the IS, a part of the IS that seemed quite trivial from a user interface perspective. Hence, it is best to be careful not to overwork Document Prototypes. In Application 1 this led to a delay of several months.

### 10.3.2 Product

#### 10.3.2.1 Overall Opinions

C1, C2 and R4 were of the opinion that the specifications were a bit too formalized. This was most obvious to C1 and C2 when working with I-Tables and Document Statecharts. They proposed to supplement each document with a verbal description of the main ideas in order to increase assimilation. However, they agreed that, in general, formality and unambiguousness in a specification is something positive. What they wanted was a smoother path into the documentation. One suggestion in this direction was to give a brief statement about how each interactive screen document was sup-

---

50 The optimization was performed just by clicking a button.

51 Calendar months that is, not man months.
posed to function, in addition to Statecharts and I-Tables. R4 did simply not feel that the extra effort required in maintaining the detail in the documentation was worthwhile. Still, R4 was of the opinion that it was important to document the system before handing it over to the maintenance organization. The VIBA documents did not work as analytic tools due to R4, which ‘probably has to do with the lack of tool support’ (R4 in I6).

Another concern brought up by several students during Application 3 is the complex handling of references between the different VIBA deliverables, such as pointers to activities in Action Diagrams from Message Definitions. This obviously relates to the current lack of integrated tool support as discussed below (Section 10.3.3).

10.3.2.2 Action Diagrams
All participants in Applications 2 and 5 showed a positive attitude towards the Action Diagrams. C2 emphasized the need to understand business routines and rules, and acknowledged that Action diagrams help to give such understanding. One problem, noted by both C1 and C2, lies in the limitations of the paper format. Lots of connectors are needed to relate different Action diagrams showing different business contexts (or parts of the same context). Of course, this would be helped by appropriate tool support.

During the early phases of Application 2, Action Diagrams were used to communicate with the users, who were non-software professionals. That approach was very successful and the users had no difficulties in following the discussions. The Action Diagrams were subsequently used as road maps while Document Prototypes were discussed and refined together with the users. This helped the users to (1) put the prototypes in familiar contexts, and (2) focus discussions on the particular interactive situation under analysis. When not relating the prototypes to the Action Diagrams, the users had difficulty in focusing on the particular task that we were interested in at the moment. The reason was that the same documents were used in more than one ISP.

One problem experienced by R1 during Application 1, however, was that iterative tasks were somewhat too complex to be visualized efficiently with Action diagrams, which are most appropriate for describing sequential flows of activities. This was also a problem in Application 2, according to both R1 and R3.

10.3.2.3 ISP Lists and Use-Situation Lists
The ISP List was ‘invented’ during the analysis phase of Application 2 as a result of problems with navigation in the documentation. The ISP List was also appreciated by C1 and C2 as giving a good overview. However, one suggestion (by C2) was to include references to Statecharts and Class diagrams. This can easily be augmented (spelled out) to the corresponding messages in the ISP List. Another suggestion was to mark each e-interaction with the number of the ISP to which it belonged. This has been incorporated in VIBA’02.

On reflection during Application 3, the term ‘ISP List’ was changed into ‘Use-Situation List’ due to the fact that the other two types of use-situation (automatic and consequential) had to be documented in a similar way to interactive use-situations.

10.3.2.4 I-Tables (E-Interaction Lists)
One major criticism regarding I-Tables was brought up by C1. The fact that all e-interactions are treated equally in the I-Tables means that trivial e-interactions are described as fairly complex. Examples of such e-interactions are simple GUI-updates such as changing one or several data fields. Such e-interactions often lead to the phrase
'Update GUI but not action memory'. This came up also during specification by R1 and R3. Perhaps it might be possible to distinguish simple e-actions from complex ones and treat them differently. This was also a concern for many students during Application 3.

One suggestion that came up during Application 2 was that the I-Tables should be presented in an order that reflects the main course of e-interactions. Both C1 and C2 had initial difficulties when assuming that such ordering was present, which it was not. However, one must remember that e-interactions constitute a set. Hence, it is not always possible to present a ‘logical’ order among them. The suggestion is, nonetheless, worth considering.

During Application 4, this idea was tried in the way that e-interaction was grouped according the ‘mode’ of interaction (that is, e-interactions belonging to the same high-level state of a document). R3 considered this to be a workable approach.

In Application 4 a somewhat simplified variant of I-Tables were used. When initially creating ‘complete’ I-Tables in accordance with their description in Section 9.7.1 two things were noted (according to R3 during I5). First, the interpretation phase of the EIAL (see Section 4.5.2) represented by the last row in the I-Tables was not considered to add anything to the understanding of the human-IS interaction. The interpretation act seemed to be ‘state fact’ in all the I-Tables and thus felt redundant. The creation of I-Tables thus became mechanical and the analysis of the interpretation act became an unreflective ‘cut and paste operation’ (R4 in I6). A similar note was made by R1 and R3 in Application 2. This does not mean that the interpretation phase in I-Tables is unimportant in general, but that the row can be omitted if it does not add to the analysis. It is, however, important not to abandon it unreflectively. Sometimes the interpretation act is important. Of course, as a competent method user, any method should be tailored to the particular needs of every given situation. It may be a good idea to investigate further how to give method support for this choice. A further ‘modification’ of the I-Tables was made during Application 4. It was soon recognized that all e-interactions in a use-situation was performed using the same interactive screen document. Hence, the document column was omitted. This is another example of competent method use, as long as it does not complicate the subsequent use of the documentation, for example, during realization. R6, who did the realization, reported, on the contrary, that the I-Tables gave good support for implementing the functionality. Hence, there was no problem in this case. However, the comment on method support for ‘modifications’ applies to this enhancement as well.

10.3.2.5 Document prototypes

Phrases used to describe the Document Prototypes were: ‘Absolutely necessary, has been used as foundation to at least 80%’ (C2 in WC2) and ‘Very valuable to tie together the other documentation’ (C1 in WC1).

However, the prototypes were also judged as devastating for creativity during systems construction, if they are rigidly followed. An interesting comment was that prototypes ‘preclude prototyping’. This should probably be interpreted as a need to allow for prototypes to change all the way through the construction phase.

During the specification (both Application 1, Application 2 and Application 4) the document prototypes were the primary ‘language’ used when talking to the users. However, one suggestion is to use ‘lo-fi’ paper prototypes initially. This relates to the previously discussed problem of prototypes becoming too complex. By using paper
prototypes, different alternatives can be sketched, and easily thrown away if unsuitable (Cooper, 1994; Graham, 1998).

10.3.2.6 Document Definitions

The Document Definitions, which, as explained above, looked radically different in VIBA’98, were of no help at all to C1 and C2. They both felt that the information was ‘formal and useless’, that is, information for the sake of information. They got the relevant understanding from other parts of the documentation. R1 and R3 felt somewhat similar about them. However, the Document Definitions typically play a more vital role early in the development process when straightening out the purpose of each document. In VIBA’02 they are also supposed to be used actively in conjunction with Message Definitions, and as a source for connecting different pieces of the documentation used in conjunction with the Use-Situation List.

10.3.2.7 Document Statecharts

C2, who was the one developing the interface code during Application 2, admitted that the Sequence restrictions (Document Statecharts) gave good support for event-driven programming. During this work he did not, however, utilize the support given by the approach as described in detail by Horrocks (1999). This was because Horrocks’ approach was unfamiliar to him, and required too much learning time to be worthwhile during this project. Nonetheless, the Statechart approach felt somewhat complex and even very simple interactive documents tended to look complicated. Also R5 who did the programming in Application 5 found the Statecharts useful in clarifying how the different e-interactions in I-tables relate.

The obstacles indicated by C2 probably had to do with (lack of) experience of the approach. At first both R1 and R3 felt that the approach was complicated. However, while working with it, it became more and more straightforward. In I6 (Application 4), R3 even gave the following comment:

‘They [Document Statecharts] have helped… most certainly useful.’ (R3 in I5)

Thus it seems that the appreciation of Document Statecharts increases with time, as one learns how to use them. As Horrocks (1999) points out, the Statechart approach requires a shift in perspective from a functional view of user interfaces to a state-oriented one. Horrocks admits that this shift can be cumbersome, which our experience thus confirms.

One aspect of Statecharts, not commented upon by Horrocks (1999), was utilized during Application 1. Interactive documents can be regarded as objects, and the Statecharts thus represent the dynamics of such objects. This could be useful to implement inheritance hierarchies of documents; a facility supported by the development tool used. However, there were problems related to describing such inherited behaviour. This is a common problem in object-orientation, as discussed by Snoeck & Dedene (1998), which resulted in us having to create, quite unnecessarily, a separate Statechart for each sub-classed document. Even though a promising approach, we have chosen not to include document inheritance in VIBA until this has been investigated more thoroughly.

---

52 The system was constructed using MS Visual Basic, which relies heavily on event-driven programming.
10.3.2.8 Class Diagrams
As indicated previously, the Class Diagrams served as a source for understanding of the business and of the system. It also served as the primary foundation for creating the relational database schema. However, using Class Diagrams, and thereby an object-oriented perspective during analysis, leads to some inherent problems when translating analysis models to a relational schema (cf. Ågerfalk, 1999a).

The structuring of Class Diagrams in one diagram per use-situation caused some initial confusion during the design phase of Application 2. However, once straightened out, this was no longer a problem. Nonetheless, according to C2, the system global Class diagram was more useful than the ‘partial models’:

‘Without [the global Class Diagram] it would have seemed very unclear… Whether it is an ISP concerning ordering or purchase doesn’t matter, we look at – what relationships there are between these classes’. (C2 during I1)

Probably, the division of the class model into parts is of more value during analysis and specification than during design and construction. R1 and R3 felt that the approach helped when creating the model. When a global class model is created from scratch, it is hard to know where to begin. It is interesting to notice also that the global Class Diagram was created automatically from the partial models by the CASE-tool used.

10.3.2.9 Class Definitions
The Class Definitions were, not surprisingly, used as the primary basis for the design of relational tables, and specifically when allocating attributes to different tables. According to C2 they were used extensively.

10.3.2.10 Missing Documentation
One interesting suggestion, made by both C1 and C2 during Application 2, was that a data model (for example, E/R-diagrams) ought to have been created during analysis. The reason for this was that some information was incorrect or missing from the Class Definitions and the Class Diagrams. The interesting thing about this is that a class model contains a superset of the semantics of a ‘traditional’ data model. Of course, foreign keys and candidate keys, for example, were not explicitly stated in the specification. However, the problems do not seem to derive from the formalism per se, but from the use of the formalism and from the consultants’ lack of UML experience. Nonetheless, some information was missing from the specification. Information that, according to C2, probably would have been identified if a deeper analysis of existing data structures had been made. The problem was thus that the conceptual model was an idealized one that did not reflect the format of data that the IS was supposed to import from other systems. This led to an analysis model with assumptions about, for example, attributes of classes, that was not suitable for storing in a ‘one class, one table’ fashion. Hence, the conclusion to be drawn is that substantial attention should be paid to existing data structures during analysis in order to ease the burden of translating the analysis results to a working design.

As a consequence of the lack of class attributes in the specification, the idea of explicitly referring to attributes by use of ONN in message definitions came up. This approach should, according to Blaha and Premerlani (1998), be a practicable way to continuously verify the class model’s completeness and consistency. The approach has been utilized in Application 3, with positive results.
During Application 4 some complementary deliverables was produced. For example, a high-level conceptual model describing sources of knowledge in the business was created to promote overview. Another type of document was an informal model showing how different screen documents relate. This was a bit surprising since high-level Document Statecharts contains precisely this information, in a more formally verifiable way. Perhaps this has to do with the Statechart learning threshold mentioned above. It may have been regarded sufficient with a less formal model, even though R3, somewhat confusedly, stated:

‘I don’t know… It’s actually a form of Statechart… It was clearer in the communication with users’. (R3 in I5)

Obviously, the latter remark vindicates that the choice of abandoning the Document Statecharts and inventing a less formal alternative was due to practical reasons and a perceived overly formalized approach inherent in them.

10.3.3 Concluding Remarks

One thing that has become obvious, if it was not so already, is the need for formal training of method users in order for them to appreciate and accept a method. Our aim of incorporating ‘standard’ modelling techniques and notations thus seems to be a promising approach. Problems related to knowledge of method and systems development skills occurred during Application 2. Of course, abstract system models are not optimal for communication with non-software professional users. This is also one reason why the new re-designed VIBA adopts a prototyping approach. However, we experienced problems that probably could have been avoided through formal training of the IT consultants and the systems co-ordinator. Both C1 and C2 have had formal training at University level in an early version of VIBA, called IBA (pre-VIBA’93), and were familiar with the principles of working with Action Diagrams. Despite that, they both experienced difficulties with some of the newer additions to the technique. They were not fully acquainted with UML either, unfortunately. Since they both had a general understanding of object orientation they managed, after surmounting some initial obstacles, to interpret the Class Diagrams, and produce a relational database schema based on those. SC, on the other hand, had no formal training in systems engineering at all, which we (R1 and R2) realized a little late. On the other hand, SC was used to working with Action Diagrams from earlier projects. He was also familiar with relational databases, but not in terms of entity/relationship diagramming (or object orientation for that matter). As a consequence, errors that we thought had been straightened out eventually ended up in the specification. And, since C1 and C2 took the specification as the ‘ultimate truth’, time consuming restructuring of the developed database was the result.

What, then, can we learn from all this? Well, first of all, formal training of all actors participating is imperative. How, and to what extent, remains to be evaluated though. Another conclusion is that the need for training and the appreciation of different modelling techniques depend to a great extent on the actors involved. For example, C1 gave a more positive image of Action Diagrams as a source of business understand-

---

53 The semantic richness of Action diagrams has continuously evolved over the years.
54 We would like to emphasize that we do not blame anyone other than ourselves for this. It was, of course, our responsibility to verify and validate the specification.
ing than C2. C2 on the other hand made it clear that he was more used to thinking in
terms of entity-relationship modelling, and consequently rated the Class Diagrams as
the best source for business understanding.

Another lesson learned is that an important issue to handle is the lack of tool sup-
port for VIBA. According to many of the participants in Applications 3 and 4, the lack
of appropriate tools made keeping documentation up-to-date in response to changing
requirements almost impossible:

‘Miss something out and you [swearword] don’t know what’s right anymore.’
(R3 in I5)

This issue has previously been identified as an important obstacle in transferring
academic research results to practitioners, so-called ‘technology transfer’ (Kaindl et al.,
2002), which the results thus confirm.

Furthermore, one of SC’s intentions, when engaging in Application 2, was to
evaluate whether VIBA could be practicable within his organization. Practicable in the
sense that non-software-experts with some modelling experience (such as SC), should
be able to create requirements specifications of high quality. The conclusion is that the
proposed techniques do not seem to lend themselves very well to such ‘end-user-
specification’. The reason is that a key to succeeding with systems development based
on VIBA seems to be a thorough understanding of the suggested modelling techniques
as well as the underlying actability concept. Something that is not gained overnight but
requires patience and commitment. This was also experienced during Application 3,
where it took some time for the students to appreciate the method. Once they had inter-
nalized it, most of them agreed that VIBA was a good support rather than an obstacle.
This is probably the case with most modelling methods (cf. Horrocks, 1999).

10.4 Overall Reflections on the Research on VIBA

This section present some reflections upon the empirical results in relation to the high-
level re-design goals of VIBA, and to the five knowledge domains that we wanted to
influence the re-design (see Section 8.2).

10.4.1 High-Level Re-Design Goals

The high-level design goals for the re-design of VIBA were to (1) make theoretical
foundations explicit and to take full advantage of their consequences, (2) design a
method that produces documentation that is communicable between developers and
other stakeholders, and (3) make use of as much existing knowledge and good practice
as possible (see Section 8.2.2).

As discussed above, actability has been made explicit in VIBA. The question re-
mains, though, whether it has been shown to be explicit in (a) information systems built
according to VIBA, and (b) information systems specifications produced according to
VIBA.

Unfortunately, for reasons discussed in Chapter 8, the first aspect has been possi-
bile to investigate only to some extent, and remains an important research topic for the
future.

In the specifications produced, the action character of the specified information
systems became visible. As one example, according to C2 (if one interprets his words
from an action perspective) the produced Document Prototypes and Document State-
charts show explicitly what e-actions and e-interactions that are possible to perform, and in what sequence. What was interesting, though, was that the Document definitions, in which the separation between propositional content and action mode was made explicit, did not attract C1 and C2. According to them, that information was visible already in the Action Diagrams. Probably, this was, at least partly, a result of insufficient internalization of actability. The separation, and hence the Document Definitions (Message Definitions in VIBA’02), should be highly important during, for example, database design (cf. Chapter 6).

As discussed above, the documentation was communicable, even though some formal training would have increased the understanding. Both the Action Diagrams and the Document Prototypes were well suited for communication with the users. The suggestion to include more ‘prosaic’ descriptions in addition to the more formalized notations used so far seems like a good idea. To what extent this is done ought to be decided during each particular development situation, and depends on the developers’ experience of (1) the method, and (2) the business under development. It is also dependent on (3) the development process model used (waterfall, iterative, et cetera) and the project setting.

The reuse of existing knowledge and good practice is most evident in the incorporation of large parts of UML into VIBA, which C1, C2 and SC considered as positive. In fact, C1 expressed that the use of method specific notation for conceptual modelling in VIBA’93 was both unnecessary and hard to learn. The specific UML parts used are class models and Statecharts. These have both proven useful and possible to integrate with the proposed action-oriented approach. The use of prototyping is an example of good-practice that we have adhered to, as are the influences from the usability field (style guides, metaphors, et cetera). The documentation forms that have not been ‘borrowed’ are those that result from the operationalization of actability and can, to a large extent, be viewed as extensions to existing formalisms. One example is the I-Tables that clearly extend Horrocks’ (1999) approach. Another example is the Message Definitions. A Message Definition can be regarded as defining a ‘sub-universe of discourse’ for conceptual object modelling.

10.4.2 Influential Knowledge Domains

The five recent knowledge domains that we wanted to influence the re-design of VIBA are business process orientation, rapid development, object modelling, usability, and software requirements management (see Section 8.2.1).

10.4.2.1 Business Process Orientation in VIBA

Business process orientation constitutes an underlying perspective of actability and VIBA. A business process can be regarded as an institutionalized pattern of activities, aiming to create value for customers. The business modelling part of VIBA (Process Modelling) builds explicitly on this view of businesses. The purpose of Process Modelling is to describe the action patterns, and the goals and problems inherent in them. A crucial part of business activities is the use of information systems, which is focused during Action Modelling and Interaction Modelling. However, one criticism that can be directed towards VIBA is that the business process perspective is not sufficiently explicit. The next step in the development of VIBA would probably be to incorporate modelling techniques that make this perspective more explicit. There are techniques focused on capturing and describing business processes at a higher level of granularity.
than Action Diagrams, and on relating business processes to each other (e.g. Lind, 2002). Such modelling is important when building systems that involve more than one business process, which has been the focus of this work.

This business process orientation is most visible in the Action Diagrams, which spanned several functional units of the business in Application 2. The biggest problem related to this was that the business was not really process-oriented. For example, R1 and R3 spent much time trying to contact the right people in different departments during Application 2. Nonetheless, the approach proved useful in at least two respects. First, the specification was created based on the work tasks of the users, which actually had an ‘informal’ process structure. Second, it highlighted some communication problems between different functions, that were important to solve, but were beyond the scope of this dissertation. Hence, a process view helps reveal problems that are difficult to discover using a traditional approach.

10.4.2.2 Rapid Development with VIBA

The work reported in this dissertation has focused more on different modelling techniques than on overall process models (framework). Much of the work on rapid development, on the other hand, has to do with such overall models. Hence, rapid development has not been emphasized as much as intended from the outset. In fact, there was a contradiction in focusing on both rapid development, and expressiveness and communicable documentation, from a research perspective. The design of the research process of Application 2 led to a waterfall development process model, even though it was perhaps not the ultimate choice if one follows McConnel’s (1996) guidelines. This, together with circumstances outside our control, led to problems that ended up with yet another software project failure. The development process model suggested for use with VIBA was introduced in VIBA’00 and is based on that of the RUP (Kruchten, 1999). In our practical work, we have not had the opportunity to actually verify how this model works, even though it seems to be working for RUP.

The conclusion is that so far it has not been verified that VIBA is actually suited for rapid development. Nonetheless, it is still very possible that it is. Further research, especially empirical research, is still needed.

When applying a rapid development approach, as suggested by DSDM (Stapleton, 1997), selection and reduction of requirements, to keep time and resources fixed, are important topics. Some work has been performed in this area by, for example, Karlsson (1998). To date, we have not made any attempt to incorporate guidelines for requirements selection (or reduction), nor have we investigated the impact of such an approach on actability. This is certainly an important research topic if VIBA is to become a true rapid development method.

During the analysis phase of Application 2, rapid development principles were applied successfully (prototyping, user involvement, et cetera). The mixed analytical and experimental approach worked well both in terms of documentation and in terms of user participation. Unfortunately, the research design made it impossible to continue with these principles throughout the realization (see Chapter 8). One big problem was that C1 and C2 considered the specification to be fixed, which considerably reduced the available design space. There is, however, nothing inherent in VIBA that prevents the use of rapid development practices throughout an entire project.

During Application 4, user interface prototyping was used extensively together with the analytic techniques suggested with VIBA, which again proved to be a workable approach (see Cronholm and Goldkuhl, 2002).
10.4.2.3 Object Modelling in VIBA

Object modelling and object orientation have substantially influenced the re-design of VIBA, even though VIBA is certainly not a true object-oriented method. One reason for this is that VIBA does not commit one to object orientation during business modelling, due to object orientation’s mechanistic view of human activity, which would contradict the underlying social action perspective. However, the work of Graham (1998) introduces fuzzy logic into descriptions of objects, which is an interesting approach, worth looking into more thoroughly. We have chosen to use object orientation during conceptual modelling so that what is proposed is an object-oriented version of ‘traditional’ conceptual modelling. However, the reason for including object orientation is not pure populism. Object orientation seems to better match a language/action approach than does traditional ‘static’ entity-relationship modelling since it views action potential, in terms of static behaviour, together with propositional contents, in terms of object data attributes, as unseparable wholes (cf. Graham, 1998).

Since we have not dealt with detailed design, and object-oriented analysis, in this work, the full potential of using object orientation has not been explored. This is an area that remains to be studied. One example of extending the ‘degree of object orientation’ is to view documents as pure objects, as explored to some extent during Application 1. This seems like a ‘natural’ approach since documents are considered as dynamic entities containing information (ae-messages) and exhibiting behaviour (action potential). However, this remains to be studied further.

Object modelling, in the form proposed by the activity Analyse Concepts in VIBA, was experienced as both positive and negative during Application 2. C1 and C2 had difficulties in understanding the Class Diagrams, due to insufficient UML experience. Another problem, expressed by C2 was that it caused confusion as to whether to build the system as truly object-oriented or not. This should not be regarded as a weakness of the specification, but as a strength. A specification that does not make unnecessary presumptions about design is of greater value than one that does. In this respect, the design space was increased.

10.4.2.4 Usability with VIBA

Following the IS use-situation model by Shackel (1984)\(^5\), the main contributions of the language/action perspective, and traditional information systems development methods, have been to the relationship between the user and the task, and to the relationship between these two and the environment (Cronholm et al., 1999). Usability, on the other hand, has emphasized the relationship between the user and the tool. Our aim has been to combine these two traditions in order to achieve a better coverage of all four components, thus moving towards the A³ model’s view on IS use (see Figure 1-4b, p. 11). By treating humans’ interaction with IT-systems as a crucial part of doing business, such coverage has been achieved. Further research is still needed, though. One important issue is, for example, to incorporate actability evaluation techniques into VIBA in a more structured way. As of now, such techniques are possible to integrate, but no ‘formal’ procedures for how to do that have been proposed or tried out in practice.

As pointed out by Gulliksen (1999) and Göransson (2001), different parts of systems development require different skills. To ensure that usability aspects are emphasized, they argue for adding a specific ‘usability designer’ to systems development

\(^{55}\) An IS use-situation consists, in this model, of four components: user, task, tool, and environment. See Figure 1-4a, p. 11.
Actability design: evolution and empirical lessons learned

projects. Even though an integrated effort is needed, different areas of expertise must probably be combined in order to achieve high quality actability. Based on this notion, the User Interface Designer was introduced in VIBA’00.

Usability requirements related to user interface design can, and have, become visible in the documentation proposed by VIBA. However, to what extent this is achieved probably depends on the people involved in each development project. One problem during Application 2 was that a further prerequisite, conforming to Microsoft’s style guides, was brought up when the analysis phase was almost ended. A consequence was that changes were made to the Document Prototypes without user involvement during the design phase. These changes were not reflected in the specifications, a fact that has led to some reflections on the last influential knowledge domain, requirements management.

10.4.2.5 Requirements Management with VIBA

Requirements traceability (RT) is important for systems maintenance and evolution, and hence for requirements management. RT can be viewed from two different angles. One viewpoint is traceability from business model through systems models to software system. Such traceability is important in order to, for example, predict software change and software change costs when re-designing or evolving a business. The other viewpoint concerns interrelationships between individual software requirements. The second view of traceability is important for understanding how changes propagate through a software system. The views are not orthogonal and both should be taken into account during RE. The first view on RT is addressed by VIBA in that requirements are derived from the I-Tables, which in turn are derived from Action diagrams that model the business. To make use of this traceability, the requirements contained in the software requirements specification must have references to the model elements to which they relate. There are approaches to the other view of RT (at least at the implementation level), such as program slicing (Weiser, 1984) and program dependence graphs (Podgurski and Clarke, 1990), which might be useful. This is an area that remains to be studied further in order to make explicit use of the business oriented action approach of actability and VIBA.

Software requirements management has not been explicitly dealt with during the empirical work. The foremost reason for that is the lack of sufficient tools for managing the different models proposed by VIBA. Because of this, changes made during the realization were not reflected in the specification, which, of course, is not good requirements management.
Part IV

Actability Evaluation
Chapter 11

Research on Actability Evaluation

As described above (Chapters 1 and 3), the research on Actability Evaluation has resulted in a method for in-context information system (IS) evaluation. This chapter describes the development of this method.

11.1 Actability Evaluation Background

As mentioned in Section 1.4.1, during the work on VIBA it became clear that there also was a need to assess existing systems as well as making formative evaluations. From a research perspective, it was important to gain empirical experience from working with the actability concept without having to develop new information systems from scratch. Therefore a research project termed ‘Actable Information Systems: Through Evaluation and Redesign’, funded by the Swedish Council for Work Life Research (RALF) and later the Swedish Agency for Innovation Systems (VINNOVA), was initiated in the late 1999. The work done on actability evaluation has been conducted within this project, which is planned to continue until the end of 2003.

11.2 Actability Evaluation Assumptions and Design Goals

Slightly modified, the first re-design goal of VIBA (see Section 8.2) also served as an important point of departure for the work on actability evaluation: conceiving information systems as information action systems is a key success factor in systems evaluation. As in the work on VIBA, the knowledge domains of business process orientation and usability (see Section 8.2) have influenced the work on actability evaluation. More specifically, actability evaluation is founded on a belief that (a) information systems use must be understood in context involving actual users (Beyer and Holtzblatt, 1998; Holmlid, 2002) while it is important (and sometimes cost-effective) to allow for expert-based inspection strategies as well (Nielsen, 1994), and (b) the perspective of evaluators inherently influences their work, which is why there is a need to explicate the perspective and work according to a set of well-defined principles. In this case, the particular perspective is that promoted by actability and the method for actability evaluation should thus be firmly founded on a set of actability principles (see Section 12.3.1).

11.3 Actability Evaluation Research Approach

The research on actability evaluation reported in this dissertation has followed the same approach as that on VIBA (see discussions in Chapters 3 and 8, and specifically Figure 8-3, p. 137). Therefore, this section will focus on differences rather than repeating what was discussed in Chapter 8. One major difference is the lack of any existing method to start out from. Instead existing approaches for usability evaluation (mainly Nielsen’s (1994) heuristic evaluation and Beyer and Holtzblatt’s (1998) contextual enquiry) and general evaluation methods (mainly inspired by Patton’s (1990) work) have been blended into an integrated approach.

The empirical work reported in this dissertation was carried out in four evaluations of existing information systems. These are referred to as Applications 5–8 (to not con-
fuse them with the VIBA Applications 0–4). Application 5 was a pilot study to try out initial ideas during an evaluation of a Swedish Internet banking system (henceforth referred to as the Internet Bank). In parallel with Application 5, another researcher working on the same project performed an evaluation of an IS used to book rooms, teachers, students and extra equipment at a university (henceforth referred to as the Booking System). Application 7 involved the evaluation of a corporate intranet (henceforth referred to as the Intranet). Finally, Application 8 involved the evaluation of a specific module in a larger administrative IS at a Swedish hospital used for scheduling appointments between doctors and patients (henceforth referred to as the Scheduling System). This author did not participate personally in the actual evaluation during Application 8. Instead two undergraduate students at this author’s department, who were interviewed after the actual project, made the evaluation. (See Sections 11.4–11.6 for details.) The approximate positioning in time of these applications is shown in Figure 11-1.

![Figure 11-1: Approximate time span of the different actability evaluation applications (A5–A8).](image)

Yin (1994) argues that the case study strategy is most likely to be appropriate for ‘how’ and ‘why’ questions, so the initial task is to clarify precisely the nature of the questions in this regard. In this research, the interest is in how actability can provide insights into why phenomena related to the use of information systems arise, and how to design IT-systems based on such understanding. ‘Phenomena’ refers to both positive and negative aspects of the design of a system in relation to the business context in which it is used. For example, a communication breakdown (such as when a user cannot perform an intended action or does not understand how to comprehend a previous action) would constitute a phenomenon; an apparently satisfactory feature in the system would constitute another (see Section 12.2 for further elaboration on this topic).

As discussed in Chapter 3, three different levels of interest constituting different units of analysis have been distinguished in this research: the levels of theory, practice and approach. At the practice level, the research is concerned with interacting users use of information systems to perform business actions within a specific type of social action context referred to as the business context. These four different units of analysis must be understood in relation to each other; for each unit the other three constitute its context. The practice level for actability evaluation coincides with that of actability design as discussed in Chapter 8. The evaluation of practices has been performed by

---

56 Further evaluations have been performed by the other project members, which have been or will be reported elsewhere.

57 Since this application did not directly involve the author of this dissertation it is to be regarded as secondary empirical data. The study has been important in the development of the evaluation method and has therefore been included in this dissertation, despite this fact.
use of an explicitly stated and evolving evaluation method. That method, *per se*, constitutes a further unit of analysis in the study, at the level of approach. For the aim of this dissertation, the interest is in properties of the method and to what extent it directs attention to, and helps to explain, relevant phenomena within the practice studied. That is, how it helps evaluators (in this case researchers) to create an understanding of the IS evaluated and the practice level. At the level of theory, the research is concerned with the concept of actability and the repercussions the development of knowledge of the other two levels has on it. Figure 11-2 depicts these levels of interest and corresponding units of analysis. (The parts that constitute the actability evaluation method are discussed in Chapter 12.)

![Figure 11-2: Units of analysis in the actability evaluation research.](image)

With respect to particular information systems, the research interest concerns properties of the systems’ design and how these relate to the concept of actability. These properties are interesting as examples of good and poor design, from an actability perspective. The interest in different individual interacting users is related to their subjective understanding of the system and its relation to the business context and the business actions performed within it. Particularly, it is important to understand different people’s comprehension of the system and its properties. The business context is interesting from the point of view that it frames the system use and the way the system and its use contribute to it. Such understanding is important to estimate the align-
ment between the system and the business it supports. Studying the actability evaluation method in use (as existing on the action level, see Section 2.3.4) is imperative to understand to what extent the method and its components aid in directing attention to and explaining actability related phenomena. Such understanding is imperative since it is a natural part of the evolution of the method (see the discussion about data and ideas in Section 8.3). In particular, attention is paid to experiences of working with the suggested evaluation process model (see Section 12.2), the use of the suggested actability principles (see Section 12.3.1) and the possible generation of new principles, the classes of the D.EU.PS. model (see Section 12.3.2), and of the layered model of action (see Section 12.3.3). Finally, the primary interest is directed towards actability as a concept; to what extent actability provides a relevant understanding of the use of the system within its business context, which is inevitable in the validation and refinement of the concept. Table 11-1 summarizes the interests in the different units of analysis.

<table>
<thead>
<tr>
<th>Unit of analysis</th>
<th>Main interest</th>
<th>Intended results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Information system</td>
<td>Properties of the system’s design</td>
<td>Examples of good and poor actability</td>
</tr>
<tr>
<td>Individual interacting</td>
<td>Subjective understanding of the system and its business context</td>
<td>Understanding of individual differences regarding the understanding and appreciation of the system</td>
</tr>
<tr>
<td>Business actions</td>
<td>What actions are performed and why</td>
<td>Understanding of the use of the system</td>
</tr>
<tr>
<td>Business context</td>
<td>The system’s support in the business context and the actor’s roles within it</td>
<td>Basis for understanding the role of the system and estimation of the alignment between the system and the business it supports</td>
</tr>
<tr>
<td>Actability evaluation</td>
<td>To what extent the method and its components aid in directing attention to and explaining relevant phenomena</td>
<td>Experience of working with the method and method evolution</td>
</tr>
<tr>
<td>method</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Researchers as evaluators</td>
<td>Apprehension of the method and its support for understanding the practice level</td>
<td>Understanding of the method’s support and incentives for method refinements</td>
</tr>
<tr>
<td>and their understanding</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Actability</td>
<td>To what extent actability provides a relevant understanding of the use of the system within its business context</td>
<td>Validation and refinement of the actability concept</td>
</tr>
</tbody>
</table>

As discussed in Chapter 3, the focus of this dissertation is on actability and its operationalizations. Therefore, the main focus with respect to actability evaluation is on the actability evaluation method as such (as an operationalization of actability), and hence on the level of approach. The way to develop the approach and to learn more about it and about actability has been to use it to study the practice level. As a consequence, the discussions in the remainder of the dissertation will focus on the evaluation method and its applicability, not on specific systems and work practices. However, as part of the research context these will be used as illustrations and points of reference.

### 11.4 About the Actability Evaluation Applications

This subsection presents the three applications briefly to frame the context for the empirical work.
11.4.1 Application 5: The Internet Bank
The Internet Bank is an online banking system with which users can perform most banking related activities: make payments, transfer money between accounts, et cetera. The evaluation focused on one use-situation, that of making payments to creditors. This activity involves the subtask of maintaining a record of pre-registered creditors (payment receivers). The evaluation was performed without involving the system owner (one of the major Swedish banks), and involved five users. Two researchers performed the evaluation, of which one (the author of this dissertation) had several years experience with the system as a user, and the other was completely new to it. The users consisted of colleagues within an academic department who use the Internet Bank to pay their bills.

11.4.2 Application 6: The Booking System
The Booking system is a system used at several universities in Sweden to handle bookings of rooms, and allocation of teachers, students and extra equipment to rooms and classes. The researcher who performed the evaluation is also a lecturer and has been a regular user of the system for approximately two years. The study focused on the use-situation that was considered most central in the Booking System: ‘perform booking’. In order to complete a booking, several measures have to be taken; including ‘find previous bookings’ performed using different search criteria.

11.4.3 Application 7: The Intranet
The Intranet is a system used corporation wide in a large Swedish manufacturing company. It is mainly used as a means to provide information to employees, but includes also facilities such as a phone book for handling contacts and a subsystem for booking rooms and extra equipment for meetings, in that sense similar to the Booking System. The study focused on the latter sub-system. See Section 13.3 for details.

11.4.4 Application 8: The Scheduling System
The Scheduling System is used at three Swedish hospitals to allocate patients to different resources (meetings with doctors, equipment, et cetera). The study performed by Kindgren and Messo (2003) focused on investigating empirically differences between Nielsen’s (1994) ten usability heuristics and the nine actability principles introduced in Section 12.3.1. For the purpose of this dissertation, the focus of Application 8 was on how the two evaluators, who are external to the project in which the actability principles are developed, experienced the principles during evaluation.

11.5 Actability Evaluation Data Collection
This section describes how the applications were selected and how data was recorded during the work.

11.5.1 Selection of Applications
Application 5, the Internet Bank, was selected based on the author’s previous experience with the system, and on the fact that it is a commonly used banking system in Sweden that it was likely to find users willing to participate in the study. Application 6, the Booking System, was selected on similar grounds; one of the researchers in the project was a regular user of the system and had already identified several problems.
related to its actability. Application 7, the Intranet, was selected based on the criterion that we wanted to examine a system that the system owner was willing to re-design based on our suggestions, should we come up with important recommendations. In this application we have cooperated with both the corporation running the Intranet and the constructor (a small IT consultancy firm in Sweden). The evaluators, without the involvement of this author, selected the Scheduling System in application 8. According to Kindgren and Messo (2003), the decision was based on a desire to evaluate, and possibly contribute to the refinement of, a system that affected many people. The IS of which the Scheduling System is part affects approx. 3000 people, directly or indirectly (Kindgren and Messo, 2003). The reason for including Application 8 in this dissertation is that it was an opportunity to have people outside of the project team use the actability evaluation method, and to learn about their experience.

11.5.2 Records

The collection and recording of experience focused mainly on positive and negative aspects of working with the actability evaluation method. In addition to this approach level documentation, a lot of practice level documentation has been created following the method as described in Chapters 12 and 13.

The main result of Application 5 was a project-internal evaluation report containing reflections on the evaluation method and its components, as well as a documentation of the actual evaluation (i.e. at approach level as well as practice level). No formal research logs were maintained. Instead notes where taken and the report evolved during the course of the application. The report contains reflections and ideas from both researchers that participated (R1 and R6, see Table 11-2 and Table 11-3).

Table 11-2: Actors providing data and ideas.

<table>
<thead>
<tr>
<th>Role</th>
<th>Application</th>
<th>Referred to as</th>
</tr>
</thead>
<tbody>
<tr>
<td>Researcher, evaluator (the author)</td>
<td>5 and 7</td>
<td>R1</td>
</tr>
<tr>
<td>Researcher, evaluator</td>
<td>5 and 7</td>
<td>R6</td>
</tr>
<tr>
<td>Researcher, evaluator</td>
<td>6</td>
<td>R7</td>
</tr>
<tr>
<td>Researcher, evaluator</td>
<td>8</td>
<td>R8</td>
</tr>
<tr>
<td>Researcher, evaluator</td>
<td>8</td>
<td>R9</td>
</tr>
</tbody>
</table>

During Application 7 a research log was maintained, which is the main approach level output from the application (in addition to practice level documentation). This log was maintained by R1 (see Table 11-2 and Table 11-3).

As discussed above, another researcher participating in the project (R7, see Table 11-2 and Table 11-3) carried out application 6. Experience from that application, which is included in this dissertation is based on project-internal evaluation material and informal discussions in project meetings.

Two interviews were performed with the evaluators (R8 and R9) at the end of Application 8. Similar to the interviews during Applications 2 and 4 (see Section 8.5), these interviews followed an interview guide. The evaluation performed in Application 8 was restricted to an ideal typical analysis (see Section 12.2), and the interview guide was therefore simplified into that shown in Figure 11-3. (The interview guide refers to
nine actability principles used during Actability Evaluation as described in Section 12.3.1). In contrast to the Actability Design interviews, the respondents got the interview guide in advance to prepare for the interview. The reason for this was that they had not had specific instructions to take notes about their experiences, and so may have needed some time to reflect. This approach was chosen to minimize the disturbance of their study and thesis writing.

<table>
<thead>
<tr>
<th>General</th>
<th>Opinion</th>
<th>How did that feel</th>
<th>Evolution over time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall judgement</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Understanding of the business</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Understanding of the system</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Actability principle</th>
<th>Positive</th>
<th>Negative</th>
<th>Applicability</th>
<th>Overall comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action Elementariness</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Recorded Action</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Action Potentiality</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Structured Action</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Irrevocable Action</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Remote Interpretation</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Delayed Interpretation</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Delayed Feedback</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Delegated Action</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 11-3:** Interview guide used during the interviews in Application 8.

A summary of the different records is shown in Table 11-3.

**Table 11-3:** Summary of collected Actability Evaluation data and its origin.

<table>
<thead>
<tr>
<th>What</th>
<th>Source</th>
<th>When</th>
<th>Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>Evaluation report</td>
<td>R1, R6</td>
<td>Application 5.</td>
<td>Written project-internal material.</td>
</tr>
<tr>
<td>Evaluation material</td>
<td>R7</td>
<td>Application 6.</td>
<td>Written project-internal material.</td>
</tr>
<tr>
<td>Research log 6 (RL6)</td>
<td>R1</td>
<td>Application 7.</td>
<td>Written chronologically.</td>
</tr>
</tbody>
</table>
11.6 Actability Evaluation Data Analysis

As discussed above, two interviews have been conducted at the level of approach (see Figure 11-2). These followed the interview guide in Figure 11-3. Similar to the work on Actability Design as described in Section 8.6, the interview guide was also used as a descriptive analytical framework to understand how the suggested actability principles worked. The rest of the Actability Evaluation research (Applications 5, 6, and 7) has focused on the ideas loop of Figure 8-3, p. 137, and has thus not been subject to any formal analysis. Rather, ideas have been incorporated in the Actability Evaluation method throughout the process. Again, this is similar to the approach taken in the work on Actability Design (see Chapter 8).
Chapter 12

Evaluation of Information Systems in Context


12.1 Introduction

This chapter outlines a method for evaluation of information systems based on the concept of information systems actability. Some fundamental assumptions and criteria believed to be essential for such evaluation are also presented. More concrete examples of working with the method are given in Chapter 13.

The suggested evaluation approach consists of two parts: an evaluation process model and three ‘analytic tools’ used during evaluation: a set of actability principles (heuristics, criteria), an analytical schema for classifying IS functionality in relation to action, and a layered model of action.

12.2 Evaluation Process Model

The suggested evaluation process model is divided into two phases. Each phase represents a different approach to the evaluation (see Figure 12-1) and consists of three stages: survey, evaluation and formulation of change proposals.

The first phase, referred to as ideal typical analysis, is inspired by goals-based evaluation (Patton, 1990) and heuristic evaluation (Nielsen, 1993). During this phase the IS is evaluated against a set of actability principles (see Section 12.3.1). The aims are to survey the possible uses of the system and to identify possible problems that, based on the principles as well as business goals, might lead to problems in real use-situations. This is a form of expert evaluation performed without involving the actual users and the results can be seen as theoretically justified change proposals.

The second phase, referred to as situational analysis, is inspired by goal-free evaluation (Patton, 1990) and contextual enquiry (Beyer and Holtzblatt, 1998). During this phase the IS is evaluated in its real use context together with its users. This evalua-
tion is more open-ended and leads to a deeper understanding of the business context and the use of the system within that context. The potential problems identified during the first phase are used to direct attention alongside a systematic search for strengths and problems not identified earlier. The results from this phase can be seen as empirically justified change proposals and as verification or rejection of the previously suggested change proposals. This evaluation work is also important to gain knowledge about the merits of the actability heuristics used and hence about the concept of activity as such.

**Figure 12-1:** The evaluation process model (Ågerfalk et al., 2002b).

Throughout the evaluation, identified strengths and problems are analysed in terms of effects and causes and classified according to a schema based on desired, existing, used, perceived and satisfactory functionality (see Section 12.3.2). This classification makes it possible, for example, to discuss who will be affected by a problem and the importance of correcting it. Throughout the evaluation, possible strengths and problems are referred to by the more neutral concept of *phenomena*.

The concept of activity gives us three levels of action at which the use of the system can be evaluated: the use-situation as a whole, the e-action, and the e-interaction. The choice of level depends on required granularity. This is important since we can choose to dig deeper into certain parts of an IS while remaining at a more shallow level in other parts.

### 12.2.1 Ideal Typical Analysis

The first measure to take during IS evaluation is to obtain a firm understanding of the business context in which the IS is used. This includes reconstructing and describing the overall action structures that form the processes of the business under study. In the suggested approach, this means to describe the actions typically performed in the business, and the results of, and prerequisites for, those actions. The resulting process description shows how all actions and use-situations fit together to form the business and to achieve the business’ goals. In addition to this ‘bird’s-eye view’, it is important to apply an actor’s view as well and describe individual tasks (use-situations) performed.
by the users. These two views together describe the business context in complementary ways.

During this phase, evaluation of the IS is performed without access to actual users. Instead, a combination of goals-based evaluation (Patton, 1990) and heuristic evaluation (Nielsen, 1993) is adopted. The evaluation is thus focused upon existing action potential provided by the system. This action potential is analysed with respect to a set of heuristics, referred to as actability principles (see Section 12.3.1) as well as business goals and anticipated individual goals. The reason to favour such an approach is that heuristics can be used to direct evaluators’ attention to key actability issues. The idea is that theoretical models of human action (e.g. Searle, 1969; 1979; Habermas, 1984; Norman, 1988; Clark, 1996) can be used to give an indication of where the users will potentially run into problems in real usage.

The results of this phase can be regarded as theoretically justified change proposals, related to an initial understanding of the business context and the actors and actions that the IS is supposed to support.

Note that at this stage it is still uncertain how the system is actually used and how the business is actually conducted. It is important to dig deeper into more hidden realms during a situational analysis. Sometimes, however, this shallow analysis is sufficient. For example, when time and other resources are limited, ideal typical (heuristic) analysis is a cost-effective method (Nielsen, 1993).

12.2.2 Situational Analysis

The situational analysis is conducted on site with real users performing real tasks inspired by goal-free evaluation (Patton, 1990) and contextual enquiry (Beyer and Holtzblatt, 1998). Goal-free evaluation means ‘gathering data on a broad array of actual effects and evaluating the importance of these effects in meeting demonstrated needs’ (Patton, 1990, p. 116). The idea is that evaluators should be open-minded and not constrained by presuppositions about what is good or bad. This is to avoid the risk of missing important unanticipated outcomes (Patton, 1990). During this phase, the description of the IS and its business context is refined and user profiles are created. Business goals as well as individual goals can be elaborated and assumptions made during ideal typical analysis can be verified and made more precise.

During this phase, evaluation can be extended to include not only what is actually in the system, but also to what users perceive the system to provide and what they think the system should provide (see Section 12.3.2). The suggested approach to situational analysis means that evaluators work closely with users to learn both the users’ tasks and the ways they use the IS. Beyer and Holtzblatt (1998) describe such contextual enquiry as a setting that tries to resemble a ‘new-on-the-job’ situation, or a master and apprentice relationship between user and evaluator. The evaluator participates in the daily work and asks questions to learn more about the whole work context. Data collection is based primarily on positive and negative aspects (see discussion on ‘phenomena’ above) encountered in the work practice. Additionally, phenomena discovered during ideal typical analysis are used to direct attention to things that might be especially related to actability.

An approach based on contextual enquiry, which is basically a combination of thinking aloud and observations, is favoured because it enables users, at least to some extent, to express tacit knowledge (Polanyi, 1983). Together with expressed thoughts and observations, evaluators have an opportunity to understand what is problematic
Information systems actability

(which can be observed from user actions) and to learn why it causes problems. The objective of situational analysis is to enable users to show and describe how actions are performed, and to express reasons for action within the actual work context. That is, to express their knowledge in action (Schön, 1983).

12.3 Analytical Tools for Evaluation

12.3.1 A Set of Actability Principles

To perform an ideal typical evaluation of an IS as suggested above, a set of principles (heuristics, criteria) to direct attention towards issues central to actability is required. Several lists of criteria focusing on user interface (UI) design and its relation to user and task characteristics have been proposed, including Nielsen’s (1994) ten usability heuristics and Shneiderman’s (1998) eight golden rules (Keinonen, 1998). In this section a set of nine complementary actability principles explicitly focused on evaluation of IT-systems as tools for business action and communication is presented. These principles can be thought of as a summary of the actability concept expressed in more operationalized terms. Along with the principles, related implications for design are discussed. The principles are derived from the concept of actability and an interpretation of the set of ten significant features of casual face-to-face conversations suggested by Clark (1996), from the perspective of actability. Of course, the set of principles is tentative and will change along with the concept of actability as such.

The actability principles are based on work on actability heuristics by Ågerfalk et al. (2002b) and actability quality ideals by Cronholm and Goldkuhl (2002). In this presentation, the principles are structured differently to emphasize the coupling to the underlying actability concept.

The ten features of the casual face-to-face situation suggested by Clark (1996) are as follows: co-presence (participants share the same physical environment), visibility and audibility (they see and hear each other), instantaneity (they recognize each other’s action at no perceivable delay), evanescence (the medium fades immediately), recordlessness (actions do not leave any record or artefact), simultaneity (participants may receive and produce at once and simultaneously), extemporaneity (actions are formulated and executed in real time), self-determination (participants determine for themselves what actions to take when), self-expression (participants take actions as themselves). Clark (1996) describes the first four of these features as related to the immediacy of face-to-face conversations, the next three as related to the medium, and the final three as related to the control of the conversation. In the remainder of this section these features are discussed from the perspective of actability; stressing that IT-system use is performance of social business action. Table 12-1 summarizes the interpretation of Clark’s (1996) features from the perspective of actability as used in the following discussion.

The comparison and interpretation (Table 12-1) is based on the notion that business communication is a special type of norm-based context (a business context) and that the introduction of the IS implies a special type of medium for the conversation.
## Evaluation of information systems in context

### Table 12-1: Features of face-to-face conversations (Clark, 1996) compared to business conversations through and by means of information systems.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Face-to-face conversation</th>
<th>Business action through IS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Co-presence</strong></td>
<td>Participants share the same physical environment</td>
<td>Participants do not share the same physical environment</td>
</tr>
<tr>
<td><strong>Visibility</strong></td>
<td>Participants can see each other</td>
<td>Participants cannot see each other</td>
</tr>
<tr>
<td><strong>Audibility</strong></td>
<td>Participants can hear each other</td>
<td>Participants cannot hear each other</td>
</tr>
<tr>
<td><strong>Instantaneity</strong></td>
<td>Participants perceive each other’s actions with no perceptible delay</td>
<td>Participants perceive each other’s actions with considerable delay</td>
</tr>
<tr>
<td><strong>Evanesence</strong></td>
<td>Medium is evanescent – it fades quickly</td>
<td>Medium is persistent – it may stay until the system is shut down</td>
</tr>
<tr>
<td><strong>Recordlessness</strong></td>
<td>Participants’ actions leave no record or artefact</td>
<td>Participants’ actions leave a record in the action memory</td>
</tr>
<tr>
<td><strong>Simultaneity</strong></td>
<td>Participants can produce and receive at once and simultaneously</td>
<td>Participants either produce or receive as separate acts</td>
</tr>
<tr>
<td><strong>Extemporaneity</strong></td>
<td>Participants formulate and execute their actions extemporaneously, in real time</td>
<td>Participants formulate and execute their actions reflectively during extended amounts of time</td>
</tr>
<tr>
<td><strong>Self-determination</strong></td>
<td>Participants determine for themselves what actions to take when</td>
<td>Business rules and IS design determine what actions to take when</td>
</tr>
<tr>
<td><strong>Self-expression</strong></td>
<td>Participants take actions as themselves</td>
<td>Participants may take action on behalf of other people and their organization</td>
</tr>
</tbody>
</table>

### 12.3.1.1 The Principle of Action Elementariness

When using an IS, communicators and interpreters are typically not co-present. When lacking co-presence, implicit references to anything outside of the system are likely to lead to confusion and misunderstandings. In the context of business IT-systems, there is typically neither visibility nor audibility. ‘Typically’ since participants may see and hear each other with the adoption of multi-media type of interfaces, even though in typical business IT-systems, such means of communication are probably rare. When participants cannot see and hear each other, gestures, sighs, and other audiovisual clues are not possible to utilize. Taking these features into account the principle of action elementariness is founded on the notion that an IT-system is a system for handling action elementary messages (ae-messages), as a result of elementary communication actions (e-actions), as suggested by actability (see Chapters 4 and 5). Ae-messages are regarded as elementary information units carrying a propositional content (what is talked about) and an associated action mode (what speaking does, an ‘illocutionary force’). To represent the social context of e-actions, ae-messages also carry information about the communicator and intended interpreters of the message. Since participants cannot see, hear or by any other physical means recognize each other or what they are ‘talking’ about, these properties must be formalized and encoded in the system; since ae-messages are regarded as the elementary information carrying units, they should be treated as such in the systems. This means that their representation in screen documents should be consistent, congruent and complete. From a design perspective the ae-
message terms (propositional contents, signifier of action modes (such as illocutionary verbs), and information about communicators and intended interpreters) should be visible and kept together in the UI. The issue of making actors visible is not uncontroversial. CSCW research points out that making work visible also open up for criticism, which may lead to more formalized work and reduction of sharing (Ackerman, 2000). On the other hand, Cronholm and Goldkuhl (2002) report on situations in which actor information is essential since interpreters need to get hold of communicators for clarification. From the perspective of actability it is important to avoid the objectification of information inherent in the descriptive perspective on information systems (see Chapters 5 and 6). This should not be interpreted as a generic requirement on information systems. The issue of visualizing actors must be treated in relation to the particular system and the important thing is to highlight the issue and not to fall into the trap of unreflective anonymization of information origin. Keeping ae-messages together means that it should be clear from the interactive documents which action mode signifier belongs to which propositional content. It also means that there should be no propositional content that does not belong to any such signifier. The principle of action elementariness also means that separate messages should be kept separate (one thing at a time); it is recommended not to mix up messages directed towards different interpreters or forcing users to perform several e-actions by one UI manipulation (unless this is made clear in the UI and that it is the way things are done).

12.3.1.2 The Principle of Recorded Action

In contrast to the recordlessness and evanescence of face-to-face conversations, one of the advantages of IT-systems is that business actions may actually leave traces in the system’s database. Utilizing this condition the principle of recorded action suggests that it is important to maintain an organizational memory of commitments and action prerequisites. This means that systems should be designed in such a way that information about performed actions, scheduled future actions and other action prerequisites are recorded and easily accessible. Based on the principle of action elementariness, the action record should be considered as a record of ae-messages (i.e. including all ae-message terms, see above).

12.3.1.3 The Principle of Action Potentiality

From an actability perspective, IT-systems can be understood as the set of e-actions they afford and support – they provide an action potential (cf. Carroll, 1996). That is, actions are only self-determined so far as the system design allows (see Chapter 4). Drawing on Habermas’ (1984) notion of validity claims raised by communicators and presupposed to be accepted by interpreters, e-actions supported by IT-systems should be not only comprehensible, but refer to the true (or inter-subjectively believed) state of the world, represent sincere intentions, and be possible to perform in accordance with socially accepted norms. Therefore the system should assist users in raising and evaluating the validity of e-actions and corresponding ae-messages. To support such a design, the principle of action potentiality suggests that it is imperative that known and understandable effects of possible actions are clearly described and that their meaning and validity are understood and agreed upon. From a UI design perspective, these conditions stress the use of expressive interactive UI components (icons, labels, et cetera). It also stresses that information that the system requires from performers should be meaningful and easily provided to the system, and that information shown should be adequate (necessary and sufficient) so that actions can be intuitively based on it.
basic condition for achieving such designs is that the language used corresponds with the users’ regular professional language. Finally, drawing on Habermas’ (1984) notion of discursive action, if uncertainties arise concerning possible actions and their validity, systems should support justification by explanations, and preferably (re-)negotiation of particular actions (at the instance level; see Table 4-3, p. 67) as well as of the action potential as such (at the type level; see Table 4-3, p. 67).

12.3.1.4 The Principle of Structured Action

While turn taking is self-determined in face-to-face settings and constrained by the very fact that any system restricts possible actions to perform (see above), business rules and norms largely determine what actions to take when in a business context. E-actions are part of an ‘action structure’ and the set of possible e-actions to perform at any particular point in time changes as the interaction proceeds. Founded on these assumptions, the principle of structured action suggests that, in order for a system to be comprehensible, users should always know what they are doing, and what they are supposed to be doing, based on what is presented in the UI. That is, possible actions to take should not only be visible (as suggested by the principle of action potentiality), but the choice of course of action to take should be informed by the system (the system could, for example, help users to live up to previous commitments made). To support users’ participation in various overlapping activities, systems should admit focus and work task changes. Still, sequence restrictions should be enforced when necessary and desirable. It is also important that the principles adopted for navigation in the system are made explicit and that the overall type of use-situation at hand is communicated (e.g. whether it is possible to perform action through the system, or if the system only provides information for use outside of it).

12.3.1.5 The Principle of Irrevocable Action

When performing actions through and by means of an IT-system, the UI is typically not evanescent, but persistent for as long as participants desire. Unless an error occurs, a displayed message will stay displayed until it is deliberately turned off. This condition gives rise to the opportunity to formulate and execute e-actions reflectively during extended amounts of time. In order to facilitate this non-extemporaneity the principle of irrevocable action suggests that IT-systems should be explicit about when an action is actually performed and provide rollback (undo) as far as possible (i.e. until a new social fact has been established).

12.3.1.6 The Principle of Remote Interpretation

The lack of co-presence (see above) also gives rise to a further principle of remote interpretation, which suggests that systems should be designed as to take into account where and how messages are received; receipt and interpretation should be possible at desired places and in desirable ways.

12.3.1.7 The Principle of Delayed Interpretation

Even though IT-systems are interactive, participants may not perceive each other’s actions instantaneously, but rather with considerable delay. Consider, for example, an order placed during the night. Such an order may not be displayed for an order-recipient until the next day. Therefore, relying on the principle of recorded action (see above), the principle of delayed interpretation suggests that keeping track of time-
Information systems actability

stamps of actions is important. It also suggests that care should be taken so that messages reach intended interpreters in due time.

12.3.1.8 The Principle of Delayed Feedback
When using an IT-system, participants either produce or receive – but not simultaneously, as in a face-to-face setting. This condition gives rise to the principle of delayed feedback stressing that users should understand that no feedback on business effects is given until a message has been delivered and interpreted. For obvious reasons, the design should strive to minimize this delayed feedback.

12.3.1.9 The Principle of Delegated Action
All communication has an ingredient of self-expression. Nonetheless, one aspect of business action stressed by actability is that people as well as IT-systems may take action on behalf of other people and their organization. This means that social obligations may be created through an IT-system without the responsible parties’ direct interaction with the system. This condition leads to the principle of delegated action which suggests that performance of actions should be allocated to human actors and information systems so that users gain maximal support (e.g. in terms of decision support vs. automated actions), that description and explanation of the system’s performed and scheduled future action(s) are readily available, and that all actors involved are aware of their action relationships, even though they may not be directly interacting with the system.

12.3.1.10 Summarizing the Principles
Table 12-2 summarizes the above presented actability principles together with major implications for IS design, relevant for evaluation as well as development of information systems.

Table 12-2: A set of nine information systems actability principles and corresponding major design implications.

1. **Action Elementariness** – An IS is a system for handling ae-messages as results of e-actions
   - Propositional content, signifier of action mode and information about communicator, performer and intended interpreter should be visible and kept together.
   - Separate messages should be kept separate (one thing at a time).

2. **Recorded Action** – Participants’ actions leave a record in the action memory
   - Information about performed actions and other action prerequisites should be recorded and easy to access.
   - Information about performer, communicator and intended interpreter(s) should be recorded and easy to access – both role and person.
3. **Action Potentiality** – an IS is the set of e-actions it affords and supports
- Known and understandable effects of possible actions should be described.
- Users should be given the possibility to evaluate the truthfulness, sincerity and normative rightness of communicated intentions.
- Expressive interactive user interface components (icons, labels, etc.) should be used.
- Information that the system requires from performers should be meaningful and easily provided to the system.
- Information shown should be adequate (necessary and sufficient) so that actions can be readily based on it.
- The language used should correspond with the users’ professional language.
- If in doubt about the action potential and its validity, support justification by explanations, and possibly negotiation, should be available.

4. **Structured Action** – Business rules determine what actions to take when
- The system should admit focus and work task changes.
- The type of navigation should be explicit.
- Sequence restrictions should be enforced when necessary and desirable.
- Performers should always know what they are doing, and what they are supposed to be doing, by only looking at the interactive screen documents available. That is, possible actions to take should be not only visible but choice of course of action to take should be informed by the system.
- Following up previous commitments should be supported by the IS.

5. **Irrevocable Action** – Actions are formulated and executed reflectively during extended amounts of time.
- The IS should be explicit about when an action is actually performed.
- Rollback (undo) should be provided as far as possible.

6. **Remote Interpretation** – Participants do not share the same physical environment
- Receipt and interpretation of messages should be possible at desired places and in desirable ways.

7. **Delayed Interpretation** – Participants may perceive each other’s actions with considerable delay
- Messages should reach intended interpreters in due time.

8. **Delayed Feedback** – Participants either produce or receive (but not simultaneously)
- Users should understand that no feedback on communication effects is given until a message has been delivered and interpreted.
- Delayed feedback on communication effects should be minimized.

9. **Delegated Action** – Performers may take action on behalf of other people and their organization
- The performance of actions should be allocated to human actors and information systems so that users gain maximal support in terms of, for example, decision support vs. automated actions.
- Description and explanation of the system’s performed and scheduled future action(s) should be readily available.
- Communicators should be aware of their action relationships.

12.3.1.11 **Actability Principles and Usability**
The suggested actability principles obviously relate to other design guidelines and heuristics, such as Nielsen’s (1994) 10 heuristics and Shneiderman’s (1998) 8 golden rules. Such heuristics, or usability principles, are typically restricted to, or at least focused on the design of user interfaces (Keinonen, 1998) and represent what was referred to as an instrumental orientation in Chapter 7. One important difference between instrumental usability principles and the socially oriented actability principles is that the former are more generally applicable whereas the latter are explicitly related to information systems as tools for business action and communication; that is, in situations when communicative rationality is the basis for understanding. As discussed in
Chapter 7, usage and, consequently, instrumentality is an important part also of communicative action. Therefore many of the usability principles apply to many of the actability principles, even though the latter are typically more focused on communication than (strictly) interaction. To illustrate some main differences and similarities, the actability principles will be related to a set of usability principles derived from the following sources of guidelines:


The set of usability principles, derived from these four sources, are the same as those discussed by Keinonen (1998)\textsuperscript{58}. Table 12-3 shows these principles and relates them to the guidelines that address them (marked as \(Ɣ\) and to specific actability principles (marked as \(Ŷ\)).

As in Keinonen’s (1998) summary, on which the following descriptions of the various usability principles are based, there is no one straight mapping between the principles and the references. The various authors have arranged them differently and some are expressed more or less explicit. It is notable that all principles are recognized by several guidelines and many of them by almost all.

**Table 12-3:** A set of usability principles frequently mentioned in the literature (with references to their sources) and their main relations to the nine suggested actability principles (see Section 12.3.1.10 for a summary).

<table>
<thead>
<tr>
<th>Usability principle</th>
<th>Source</th>
<th>Related Actability Principle</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1 2 3 4</td>
<td>1 2 3 4 5 6 7 8 9</td>
</tr>
<tr>
<td>1. Appropriate Presentation</td>
<td>(ƔƔƔƔ)</td>
<td>(ŶŶ)</td>
</tr>
<tr>
<td>2. Consistency</td>
<td>(ƔƔƔƔ)</td>
<td>(ŶŶŶ)</td>
</tr>
<tr>
<td>3. Error Handling and Recovery</td>
<td>(ƔƔƔƔ)</td>
<td>(ŶŶŶŶŶ)</td>
</tr>
<tr>
<td>4. Flexibility</td>
<td>(ƔƔƔƔ)</td>
<td>(Ŷ)</td>
</tr>
<tr>
<td>5. Guidance and Help</td>
<td>(Ɣ)</td>
<td>(Ŷ)</td>
</tr>
<tr>
<td>6. Memory-Load Reduction</td>
<td>(ƔƔ)</td>
<td>(Ŷ)</td>
</tr>
<tr>
<td>7. Task Match</td>
<td>(ƔƔ)</td>
<td>(Ŷ)</td>
</tr>
<tr>
<td>8. User Control</td>
<td>(ƔƔ)</td>
<td>(Ŷ)</td>
</tr>
</tbody>
</table>

Appropriate visual presentation is a concern that has arguably so far dominated the research on user interface design (Keinonen, 1998), and by implication research on usability (Holmlid, 2002). It is emphasized that users have to be provided with sufficient and not inadequate (irrelevant or rarely needed) information. This overall principle also includes the utilization of feedback of user interface manipulations and thus visibility of system status. Certainly, the appropriateness of the visual appearance is

\textsuperscript{58} Keinonen (1998) refers to four additional guidelines and uses earlier references to Nielsen and Shneiderman. The suggested usability principles are still the same.
important for actability too. Specifically, the actability principles stress some issues that contribute to such appropriateness, from an actability perspective. The principle of action elementariness, for example, suggests what is appropriate based on the concept of the ae-message. The principle of action potentiality stresses that the visual appearance should aid users in evaluating and raising claims of validity, that expressive UI components clearly describing the action potential are used, and that adequate information in relation to users’ actions is presented. Furthermore, the principle of delayed feedback points out that feedback exists on two different levels and that both are important. The first level is that of ‘interaction feedback’ typically stressed by usability principles. The second level is that of ‘business feedback’ related to the fact that business effects may not occur instantaneously. Both types of feedback should be acknowledged and taken into account when evaluating an IS.

The usability principle of consistency refers to interface designs that keep to the same principles over a set of individual systems or parts thereof. This essential design principle addresses a wide spectrum of topics including the use of terminology, interaction patterns and consistency between applications. It is claimed that consistency facilitates learning since re-learning is minimized when people already know their way about. Visual consistency also increases the perceived stability, thus promoting user confidence. With respect to the actability principles, consistency is important not only in relation to what is visible in the systems but also in relation to the overall communication patterns forming the business tasks and processes in which a system is used. Specifically, consistency is important to uphold the principles of action potentiality and irrevocable action, and the design ideals suggested by the principle of structured action should be applied consistently to make them effective. One specific point relates to the principle of action potentiality. The same e-action may be performed in different ways in different parts of a system, or outside of it as manual action. From a design perspective it is important to make clear that these alternatives actually concern the same business action.

All of the referenced usability principles include aspects of error handling and recovery (sometimes referred to as ‘forgiving the user’). Error handling includes warnings given by the system before hazardous commands, information about irrevocable user actions, error prevention strategies, detection of errors made, easy reversal of actions, and possibilities for the users to correct errors without re-performing the whole task. Error recovery is claimed to relieve anxiety and facilitate learning by doing (i.e. ‘trial and error’). The actability principles do not explicitly address error handling and recovery, in an instrumental way, but many of them are intended for minimizing confusion and the potential of performing erroneous or unintended business actions. For example, the principle of action elementariness aims at preventing confusion and errors (which, if performed, may not be tracked until it is too late, due to the principle of delayed feedback). Another important issue is that of actor visibility, stressed by the principle of recorded action, to sort out potential problems before they occur, a sort of ‘social error handling’ or at least prevention. This is also the case with the principle of irrevocable action stressing that systems should be clear about when an e-action has actually been performed, thus preventing ‘business errors’ such as unintended business commitments. Similarly, the principle of action potentiality may decrease error rates since it focuses on delivering a correct and understandable action potential to users. Furthermore, the proper use of sequence restrictions proposed by the principle of structured action is a way of avoiding errors related to both interactions (leading, for example, to reworking at the interface) and business actions (leading to ‘business errors’).
The usability principle concerned with flexibility suggests that different interaction strategies should be allowed for different (types of) users. For example, accelerators and shortcuts hidden for inexperienced users may speed up the interaction for expert users. Further, the principle suggests that users should be allowed to tailor frequent actions, and to ‘personalize’ the system. These important types of flexibility are typical instrumental concerns and are consequently not stressed by the actability principles. However, the actability principle of structured action points at a need for grounding flexibility in business goals and rules. Since we are dealing with finite automata, there is a need to consider the potential trade-offs between flexibility and structure.

According to the principle of guidance and help, informative and relevant user support should be provided, both interactively (on-line help) and in printed documentation (user guides and manuals), to help the user understand the system. What actability adds to this issue is the possibility of designing systems that allow not only for explanation but also negotiation. In many systems this is probably utopian today even though the ‘personalization’ mentioned above can be seen as a related topic.

The notion of memory-load reduction addresses a basic principle of human cognition: People tend to not remember unrelated pieces of information accurately. It is expected that many errors will occur where precise recollection is required, which is why interaction should rely more on recognition than recall. It is argued that recall is prone to error, while people are good at recognising objects. The system should therefore present alternatives and patterns so that people can select among given options. This usability principle goes hand in hand with the actability principle of action potentiality, which emphasizes that information should be meaningful and easily provided, for exactly these cognitive reasons. This is also a reason why the ‘business effect feedback’ should be minimized as suggested by the principle of delayed feedback.

According to the notion of task match, information systems should present exactly the information that users need. The information should be presented in the way the user prefers to utilize it. Many different expressions, such as right mappings, compatibility, exploitation of constraints, and identity cues, refer to this property. This topic is pretty much what actability is altogether. It is by understanding user tasks as part of communicative business structures that actable systems are possible to conceptualize. From a systems perspective, the principle of structured action also suggests that an IS should support task changes, which specifically relate to task match.

The idea of supporting users’ internal locus of control relates to the user’s subjective feeling of first person participation and engagement in the interaction. It is also related to the design principle of direct manipulation interfaces. The suggestion is that interaction is more subjectively rewarding if users feel that they can influence the objects themselves rather than giving the system instructions to act. This relates highly to the need to make users understand the rationale behind sequence restrictions as well as to the allocation of actions between users and systems. It is thus imperative that users understand what the system is doing, possibly on their behalf, and possibly directed to other people.

12.3.2 The D.EU.PS. Model

As discussed in Chapter 7, usability, according to ISO 9241 (ISO 9241-11, 1998), is concerned with the effectiveness, efficiency and satisfaction with which users can achieve specified goals in specified contexts of use. If applying these categories unreflectively, then an implicit assumption is made that information systems consist of
desired (effective) functionality that, to varying degrees, is efficient and satisfactory. That is, there are two primary overlapping classes of functionality, desired and satisfactory, with functionality belonging to any of the classes being more or less efficient. To fully understand the use of an information system (IS) and to assess its usage quality there seems to be a need to introduce further categories of IS functionality. Therefore, the D.EU.PS. model, which is used to discuss IS functionality within the proposed method for actability evaluation, consists of eighteen classes derived from the combination of five high-level categories of IS functionality. To understand these classes and categories, let us take our point of departure in a straightforward model of action spaces proposed by Berglind (1990).

In any specific situation, an actor may or may not act depending on how the situation is perceived. According to Berglind (1990), there are two basic considerations when a person is to decide whether to take action (including inaction) or not. Either you can do something, given the particular circumstances, or you cannot, and either you want to do something or you do not. If you can do what you want to do, then everything is fine. Sometimes, however, people desire things they cannot achieve, they want but cannot. You may also occasionally act because you feel that you have to, even though you may not really want to. On the other hand, sometimes you may choose not to act despite your will to do so because the action involves something you believe you should not do. These two dimensions (can and want) together span an action space, and their combinations form incentives for action or non-action, as illustrated in Figure 12-2. The grey fields in the figure illustrate that you do not always know what you want and can.

<table>
<thead>
<tr>
<th>Can</th>
<th>?</th>
<th>Cannot</th>
</tr>
</thead>
<tbody>
<tr>
<td>Want</td>
<td></td>
<td></td>
</tr>
<tr>
<td>?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Does Not Want</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 12-2: Action space as a combination of can/cannot and want/do not want (Berglind, 1990, p. 34).

The action space model of Figure 12-2 gives rise to questions such as: What possibilities do I have to perform actions? What are my intentions? What restricts me from achieving my goals? What forces me to do things that I do not want to do? In the context of information systems, these questions must be considered with an understanding of the information system as an important part of creating the action space. When action is mediated by an IS (Kuutti, 1996; 1999), two interrelated contextual aspects, the function and the form of the IS, must be considered as they together shape the action potential delivered by the IS (Ehn, 1995).

If there is something that you want to do with the IS, then there is a desired functionality. On the other hand, things that a user does not want constitute undesired functionality. The can-dimension in Figure 12-2 assumes that the functionality is perceived in use – that the user sees and understands the action possibilities. This dimension gives rise to questions such as: What action possibilities, and what restrictions, does the user experience when using the IS? In order to believe that you can use a function, you have to perceive that function. A function perceived as something you cannot do with the IS is something experienced as a restriction – an action potential that the IS does not offer.
On the other hand, perceiving something as undoable does not necessarily mean that you do not perceive the corresponding functionality as such. It may, for example, be the case that you cannot use a function since you do not know how to use it.

Ehn and Löwgren (Ehn, 1995; Ehn and Löwgren, 1997) emphasize that ‘form’ is often misleadingly used to mean an artefact’s physical appearance alone, and stress that form is not a property of an information system but of the relation between system and user. This view accords with Gibson’s (1977; 1979) concept of affordances. Gibson argues that in order to use, for example, a door, the actor has to perceive its properties, whether the door, for example, affords push or pull – is ‘pushable’ or ‘pullable’. Affordances are properties in relation to actors. They exist in external objects, but they are not mere physical properties – they emerge only in relation to an actor. The central issue is not whether certain affordances exist, but if they are possible to perceive in relation to oneself and one’s intentions to act. A cup is graspable if its size matches that of the person’s hand who attempts to grasp it, it is neither the hand nor the cup that constitutes this affordance, but their combination (Bærentsen and Trettvik, 2002).

Building on the notion of affordances, artefacts can be said to have intrinsic properties of form and function but it is only in use that these properties can be evaluated. The properties are expressed in use. In the design process, an information system may have been given properties to promote certain effects, for example, that the user will perceive it as effective and efficient in use. These properties are features of the information system (as they are results of the design process). On the other hand, whether a property gives rise to intended effects can only be evaluated with respect to the relation between system and user (effects emerge in use). It can result in an intended or unintended effect; the user can perceive a property differently than what was intended during design. As Norman (1999) points out, the art of the designer is to ensure that desired, relevant actions are readily perceivable.

It is also possible to live under the false impression that certain functionality exists when it in fact does not. A user may, for example, believe that she uses a function when she in fact does something else than what she thinks she does. In that case, from a designer point-of-view, the user utilizes functionality that she does not perceive. This seemingly controversial position can be understood based on Aronsson (1990) who distinguishes between subjective and objective action space. The subjective action space is what you believe you can do, and the objective action space is what is really possible to do. In practice, the distinction is not so easy to make. The relationship between subjective and objective is characterized by movement. Each individual changes his or her objective action space through, for example, increased knowledge, and so may increase his or her subjective action space accordingly. Thus, it is the perception that controls which actions you believe you can do. There may exist action space that is not perceived and perceived action space that does not exist. Of course, there is also action space that is both perceived and existing.

If applying this argumentation to information systems design, the objective corresponds to what is in the system and the subjective is what users perceive and designers intend. We may therefore distinguish the categories existing functionality from that which users perceive to exist. It is important to recognize not only what functions the IS implements but also the actions that the user perceives as possible to perform. We may also distinguish between the desired functionality from the functionality that does eventually exist in the system. The categories desired and existing may not overlap completely. There may be actions that users want to perform but cannot because the desired functionality is not implemented. Furthermore, it is a well-known fact that
many systems or parts thereof are not used to the extent anticipated during design (e.g. Davis, 1989). It is therefore essential to additionally distinguish between what exists, perceived or not, and what is actually utilized. Davis (1989) also points out that it is important not to confuse desired functionality with satisfactory functionality. Even if users believe an application to be useful (as in desired), they might still consider it too hard to use (as in unsatisfactory). On the other hand, people may perceive an undesired function as satisfactory: ‘sure, it works fine, but I don’t need it’. Utilization of existing functionality is thereby, in addition to desired functionality, influenced by perceived ease of use (as in satisfaction). Satisfaction thus relates to the like or dislike of the actual implementation of a function while desired functionality regards the desirability of a function holistically.

Altogether, this discussion leaves us with five high-level categories of IS functionality: Desired (D), Existing (E), Utilized (U), Perceived (P), and Satisfactory (S). Figure 12-3 shows how these categories relate to the action space model of Figure 12-2.

<table>
<thead>
<tr>
<th>Can</th>
<th>?</th>
<th>Cannot</th>
</tr>
</thead>
<tbody>
<tr>
<td>E P</td>
<td>E ~P</td>
<td>~E P</td>
</tr>
<tr>
<td>Want</td>
<td>D</td>
<td>Situation 1</td>
</tr>
<tr>
<td></td>
<td>¬D</td>
<td>Situation 2</td>
</tr>
</tbody>
</table>

Figure 12-3: Action space as utilized and non-utilized functionality. Capital letters refer to the categories introduced above and the symbol ¬ represents Boolean not; that is: D = desired, ¬D = undesired. (Eliason and Ågerfalk, 2003)

From Figure 12-3 we see that in Situations 1 and 2, if a user perceives (P) an existing function (E), that function can be utilized (U) even though it may not be (~U), regardless of the user’s will to use it (D) or not (~D). Whether used or not, the function may be considered satisfactory or unsatisfactory (S or ¬S). In Situations 3 and 4 we can see that even if a user does not perceive (~P) an existing function (E), that function can still be utilized (U or ~U), the user, in that case, is not aware of using it and so it cannot be satisfactory (~P ∨ ¬S). Still, it may be a function that the user desires (D) or does not desire (~D). All these situations (1–4) involve the use or non-use of existing functionality – functionality that can be used, functionality that the designer put into the system and that users may or may not perceive. This is different from the remaining Situations 5–8, which all constitute situations in which the particular functionality cannot be used (~U). If we look at situation 5 and 6, the user perceives (P) a function that does not exist (~E). The user believes that it exists, desired or not, but obviously he or she does something else than what he or she believes. The subjective understanding of the user may diverge from that of the designer. Nonetheless, since perceived, the function can be considered satisfactory (S) or unsatisfactory (¬S). The final Situations 7 and 8 constitute non-existing (~E) functionality that the user does not perceive (~P).
This can be functionality that the user desires (D), hence a missing user requirement, or a function that the user does not want (¬D).

An obvious goal of a design is that the users’ perception of the system’s functionality corresponds with that intended (Norman, 1988; Orlikowski and Gash, 1994). That is, making the subjective understanding of the designer match that of the user so that intersubjectivity is reached and maintained. Through analysing users’ perceived action space, the designer can identify and implement missing user requirements (desired non-existing functionality). Additionally, such analyses may assist in making users aware of functionality that has so far been used unknowingly. This is where the D.EU.PS. model comes into play.

The D.EU.PS. model is a comprehensive model of IS use and represents combinations of the five high-level categories of IS functionality: Desired, Existing, Utilized, Perceived, and Satisfactory. Desired functionality is functionality that is believed to enable a user to accomplish intended effects. Existing functionality is functionality that is implemented and accessible in the system. Utilized functionality is a subset of the existing functionality; it constitutes the functionality that is actually being used. Perceived functionality is functionality that users perceive to exist. Finally, Satisfactory functionality is that perceived functionality which is considered easy to use. The model (see Figure 12-4) provides a means to classify the functionality of a system into these five different but overlapping categories, constituting eighteen classes.

Note that the respective sizes of the classes in the figure do not correspond to any quantitative empirical measure. They have been chosen simply to facilitate presentation. Note also that since utilized is regarded as a subset of existing, and satisfactory as a subset of perceived, all five letters are not used for all classes in the model; the classes can be uniquely identified anyway. This subset constitution of the model is also the reason for calling it D.EU.PS.

Figure 12-4: The D.EU.PS. model (adapted from Ågerfalk et al., 2002b). As in Figure 12-3, capital letters refer to the corresponding category and the symbol ¬ represents Boolean not; that is: E = existing, ¬E = non-existing.
The resulting 18 classes as combinations of the five high-level categories with examples of each as derived from the three applications in which the model has been used as an analytic tool (see Chapter 11) are shown in Table 12-4.

The D.EU.PS. model captures important aspects to consider when studying user interaction with information systems. The model makes it possible to discuss the functionality of a system in terms of what is desired and what is not, what exists and what is missing, what is actually utilized and what is unneeded, what is believed to exist, and what can be used with satisfaction? This in turn makes it possible to identify and discuss different users’ perceptions of and attitudes towards a system, and how these perceptions and attitudes change over time, as evidenced in the examples in Table 12-4. Taking these aspects into account is important in order to pinpoint misunderstandings and focus on real usability problems. It would, for example, not be very effective to redesign a part of a system if that part is unlikely to be used anyway – often the user interface is not the problem (Göransson et al., 1987; Mathieson and Keil, 1998): ‘When EOU [ease of use] problems are observed, consulting an interface expert might not be effective if task/technology fit is really the issue’ (Mathieson and Keil, 1998, p. 227).

Table 12-4: Classes of the D.EU.PS. model (Ågerfalk and Eliason, 2003).

<table>
<thead>
<tr>
<th>Class</th>
<th>Description and example</th>
</tr>
</thead>
<tbody>
<tr>
<td>¬D ¬E ¬P ¬S</td>
<td>Users believe that this undesired function exists but that it could be better, but it does not exist at all, and we better leave it that way. This is probably related to learnability or misleading instructions/education. Example: the Intranet does not log user activities to the extent feared by some users.</td>
</tr>
<tr>
<td>¬D ¬E S</td>
<td>Users believe this undesired function exists and that it works fine, but it does not, and we better leave it that way. This is probably related to learnability or misleading instructions/education. Example: some Intranet users did not care that the system logged their activities (which it did not).</td>
</tr>
<tr>
<td>¬D E ¬U ¬P</td>
<td>This function is neither desired nor used, and not even recognized. It can probably be removed. Example: An Internet Bank user did not know that it is possible to categorize pre-registered receivers, and he did not want that function anyway.</td>
</tr>
<tr>
<td>¬D E ¬U P ¬S</td>
<td>An undesired function that the users do not need and think is unsatisfactory. Example: The function in the previous example was not only undesired but was also considered hard to use by some users.</td>
</tr>
<tr>
<td>¬D E ¬U S</td>
<td>Sure it works fine but users do not need the functionality. Example: In the Booking System it is possible to create diagrams showing bookings made by a teacher. A function that users did not want and did not use.</td>
</tr>
<tr>
<td>¬D U ¬P</td>
<td>This function is used even though users are not aware of its existence, and they do not want it. This may be a result of unattended action (e.g. trial and error) and should, in that case, be avoided. It may also be the case that the system does something automatically without the user’s knowledge and approval. Example: The Internet Bank shows a list of ‘new offers’ whenever someone logs on to the system, which may be more annoying than informative. The same goes for logging of user actions by use of cookies that the user may not be aware of.</td>
</tr>
<tr>
<td>¬D U P ¬S</td>
<td>Not only forced to use it, it is unpleasant as well. Example: Some Internet Bank users regarded the repeated entering of security codes as a waste of time.</td>
</tr>
<tr>
<td>¬D U S</td>
<td>It works well even though users do not really want it. Example: Some Internet Bank users thought that the system required to many safety codes that did not promote trust, yet they accepted it and did not think it was a big deal.</td>
</tr>
</tbody>
</table>
Information systems actability

D ¬E ¬P This should really be there. A missing user requirement. Example: An Internet Bank user wanted a function showing how much money that would be left on his account should he fulfil the current payment.

D ¬E P ¬S Users want this function and believe that it exists even though it could be better, but obviously they do something else than what they think they do. Example: When discovering that the Booking System did not book the room as first believed, the user was no longer that happy with the function.

D ¬E S Users want this function and believe that it exists and that it looks good, but obviously they do something else than what they think they do. Example: A Booking System user was very happy that he had booked a classroom, but when the class started another teacher claimed the room. The user was not aware that the system used the wrong database for storing the bookings of that particular room.

D E ¬U ¬P Users do not perceive this function so they do not use it, but they would had they understood it. Example: An Internet Bank user did not know how to change a partially completed payment order. The function was ‘hidden’ in the design since it has the same visual appearance as help texts and therefore was perceived as such.

D E ¬U P ¬S It’s so bad users cannot really use it even though they would like to. Example: A user thought that the Internet Bank’s online help did not help at all. As a consequence, he did not use it even though in need for help.

D E ¬U S Sure, but another way works even better. Example: In the Internet Bank there are two different ways to input the receiver of a payment; the other way was preferred.

D U ¬P Users perform this function even though they are not aware of its existence. Probably a result of unattended action. Relates probably to learnability or education. Example: An Internet Bank user was not aware that when a date for payment was not given, the today date was automatically used.

D U P ¬S This is good but could be better. Example: An Internet Bank user wanted to be able to register new receivers in a separate window.

D U S This is good! This is the way all IS functionality should be like! Example: An Internet Bank user was happy with the way she could check when the money had been withdrawn from her account.

A final note regards functionality that is undesired (¬D), non-existing (¬E), and unperceived (¬P); thus corresponding to Situation 8 in Figure 12-3, p. 221. This would, for example, be the case if a new feature were proposed in response to identified problems during evaluation (i.e. a change proposal) and we were to decide whether to implement it or not. That is, to decide, for example, if it is desired or not. In set theoretical terms, we can think of such a class as the complement of the union of desired, existing and perceived functionality. That is, that (possible) functionality which is not part of any of the D.EU.PS. classes, given a certain system. The D.EU.PS. model can thus be used to discuss alternatives and ‘new’ functionality in terms of ‘newness’ (does it in fact already exist, even though maybe not perceived?), as well as ‘desirability’ (is this function really desired?), and ‘perceivedness’ (if we were to implement this function, how would we make it readily perceivable to users?).

12.3.3 A Layered Model of Action

The actability understanding of action in relation to information systems makes it possible to distinguish between different levels of action (see Section 4.5). First of all, it is possible to talk about single e-actions involving (at least) a communicator, a mediator (the IS) and an interpreter. E-actions can be analytically grouped into use-situations...
consisting of coherent sets of actions. In this perspective it is important to consider not only the performer and communicator of an action, but to take into account also the intended interpreter(s), who can be performers in another use-situation. This way use-situations are linked into workflows constituting the business processes in which the IS is used. E-actions can also be decomposed into a partially ordered set of human-IS interactions performed to formulate the ae-message and execute the corresponding e-action.

This gives us three levels of action at which the use of the system can be evaluated. The choice of level depends on the granularity that is of interest. In order to decide the level of evaluation the evaluator first has to decide if the use-situation is desired, and perhaps already satisfactory (see Section 12.3.2). If order management, for example, is considered undesired it is an indication that there is no need to dig deeper into that use-situation. Otherwise, the remaining levels of granularity, that is, e-actions and e-interactions, can be analysed in terms of D.EU.PS. This is important since we can choose to dig deeper into certain parts of an IS while remaining at a more shallow level in other parts.

12.4 Documenting the Evaluation

In order to document the evaluation, many of the documentation forms (deliverables) used within VIBA (see Chapter 9) can be used. In a sense, evaluation can be thought of as requirements re-engineering. As a consequence, the activities of VIBA can be used, slightly modified, also to understand actability evaluation.

One thing that distinguishes (formative, post implementation) evaluation from development is that the IS is now a source of knowledge rather than the product of the process. This also means that the different activities may be performed in a radically different order during actability evaluation compared to that suggested by VIBA. It may, for example be obvious to start with existing interactive documents and reconstruct the e-actions possible to perform through these and the corresponding e-actions, rather than to start with high-level process descriptions in Action Diagrams to eventually arrive at the screen documents. Nonetheless, Action Diagrams are important tools in reconstructing business processes based on the understanding of a system and its use gained through analysis of possible interaction through available documents.

Depending on the level of evaluation (use-situation, e-action or e-interaction) the required analytical tools vary, and, as a consequence, the required forms of documenting analyses vary. It would, for example, not be possible to create I-Tables if the evaluation did not concern e-interactions. If I-Tables were created, analysis would, by definition, be at the level of e-interaction. The important thing is to view the suggested techniques in VIBA as a toolbox and choose the tools relevant for the evaluation and situation at hand.

When performing evaluation, different types of protocol may be used to note observations and comments. Specifically, in the empirical work with the evaluation method, protocols explicitly based on the actability principles and the D.EU.PS. model have been used (see Table 12-5 and Table 12-6).

One specific form of documentation not used within VIBA but important for actability evaluation is the Change Proposal. A correct use of change proposals would be to order them by significance and implement as many as are feasible, then conduct another evaluation to learn more about the system and the evaluation approach as such.
Table 12-5: Structure of protocols used during evaluation to note observations with respect to specific IS functions and classify them according to the D.EU.PS. model. (adapted from Eliason and Ågerfalk, 2003)

<table>
<thead>
<tr>
<th>Function Name of function and/or corresponding supported action</th>
<th>Desired Classification with comments</th>
<th>Existing Classification with comments</th>
<th>Utilized Classification with comments</th>
<th>Perceived Classification with comments</th>
<th>Satisfactory Classification with comments</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Phenomenon</strong></td>
<td>Verbal description of observed phenomenon justifying the classification and hinting about possible redesigns.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 12-6: Structure of protocols used during evaluation to note observations with respect to specific e-actions, used together with corresponding Message Definition.

<table>
<thead>
<tr>
<th>Principle</th>
<th>Phenomenon</th>
<th>Effects</th>
<th>Importance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action Elementariness</td>
<td>Verbal description of observed phenomenon.</td>
<td>Observed or anticipated consequences</td>
<td>Estimate of importance of correcting</td>
</tr>
<tr>
<td>Recorded Action</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Action Potentiality</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Structured Action</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Irrevocable Action</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Remote Interpretation</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Delayed Interpretation</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Delayed Feedback</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Delegated Action</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

12.5 A Summary of the Proposed Evaluation Approach

This chapter has outlined a contextual IS evaluation approach based on the concept of information systems actability. Actability gives a theoretical foundation for studying information systems within their business context. This is important in order to see how the use of the IS, including the distribution between IT-supported and fully automated tasks, affects and is affected by the social context surrounding it.

The suggested approach to IS evaluation combines ideal typical evaluation based on a set of principles with situational evaluation influenced mainly by contextual enquiry. The combination makes it possible to balance expert-based evaluation and in-depth participative studies taking trade-offs such as allotted time and resources into account.

The heuristics (principles) used are based on the concept of actability, and they represent a way of operationalizing the concept into tangible properties of the IS and its surrounding business context, including interacting users and, for example, system owners as providers of action potential and communicators of information upon whose commission interacting users act.
In addition to heuristics, the approach makes use of a model for classification of IS functionality (and hence action potential) as combinations of desired, existing, used, perceived and satisfactory functionality (the D.EU.PS. model). Based on that classification, causes and effects can be analysed in more detail and proposals for change formulated accordingly.

When applying the evaluation approach, a choice is given regarding on what level of granularity to perform the evaluation. Based on the concept of actability we can distinguish between three levels: the use-situation, the elementary action, and the elementary interaction performed to formulate and execute elementary actions.
Chapter 13

Actability Evaluation: Evolution and Empirical Lessons Learned

This chapter presents lessons learned from working with information systems evaluation following the method presented in Chapter 12. In the chapter, a discussion about the evolution of the method and its relations to the underlying concept of actability is presented.

13.1 Evolution of Actability Evaluation

Our work on actability evaluation has not been going on as long as the work on VIBA. Therefore, the actability evaluation method presented in Chapter 12 is the only externalized version available. A slightly different version was presented by Ågerfalk et al. (2002b), however. The difference consists in the set of actability principles (see Section 12.3.1), which Ågerfalk et al. (2002b) structured differently and termed ‘actability heuristics’. The empirical material used by Ågerfalk et al. (2002b) was Applications 5 and 6, the Internet Bank and the Booking System. As discussed below (Section 13.3.2), the heuristics changed during these applications. Application 7 has not added any new principles but indicates that the existing ones are usable. In all other respects, the presentation given in Chapter 12 is an extension of the previously published paper, and the method represents what has evolved in the project ‘Actable Information Systems: Through Evaluation and Redesign’ since its inception in late 2000 (see Chapter 11).

13.2 From Actability to Actability Evaluation

The actability evaluation method was described in Chapters 1 and 3 as an operationalization of actability. In this section major relations between the method and actability are commented upon, in order to clarify how the operationalization has been made.

13.2.1 Grounding of the Actability Principles

The actability principles (see Section 12.3.1) are explicitly derived from the concept of actability. Therefore, there is a straightforward mapping between the principles and the underlying concept. The principle of action elementariness builds explicitly on the concept of the action-elementary message (ae-message) as discussed in Chapters 4 and 5. The principle of recorded action is founded on the concept of an action memory as an important part of an IS (see Sections 4.5.1 and 4.5.4, further elaborated in Chapter 6). The principle of action potentiality is founded on the actability way of conceiving information systems as the set of elementary business actions they afford and support (see Chapter 4). The principle of structured action is based on the notion that information systems are used in business contexts that are, at least partly, organized into action structures forming the business processes of the organization. The principle of irreversible action takes into account actability as a way of conceiving human-IS interaction as formulation and sending of ae-messages (see Sections 4.5.1 and 4.5.2). The principles of remote interpretation, delayed interpretation, and delayed feedback are all derived from social perspective on information systems promoted by actability – informa-
tion systems are used as tools for communication, often involving at least two individuals (see Chapter 4, 5, 6 and 7). Finally, the principle of delegated action is based on the recognition of information systems as performers of business action on behalf of organizations and human actors (see Chapter 4).

13.2.2 The D.EU.PS. Model and Actability

The D.EU.PS. model and its classes are not specific for actability. Rather, they can be used to understand the use of any system regardless of underlying perspective. However, there are clear mappings from actability to the model. First, since actability to a large extent is based on speech act theory, it assumes that there exist intentional actions, performed by rational actors. Hence, there are e-actions and corresponding IS functionality invoked to perform these that conforms more or less to the actor’s intentions – there is more or less desired functionality. As explained in Chapter 4, actability also draws on the notion of affordances, that a system is what it is perceived to be. Hence, there is perceived and unperceived functionality, which may be regarded as more or less satisfactory. Furthermore, actability is founded on the ontological standpoint that there is an external reality independent of human actors’ interpretations of it but that actor’s subjective and inter-subjective understanding of the world is the basis for actions (see Section 3.2.1). Hence, there are intrinsic properties of behaviour in information systems. Such properties may be perceived as existing functions. There is also possible functionality (in people’s minds) that does not exist in any IS. Potentially, what exists may be used.

13.2.3 Actability and the Layered Model of Action

The layered model of action stems from the actability decomposition of actions in IS use-situations into e-actions and e-interactions as discussed in Section 4.5.2. These two levels of granularity are chosen based on the notion that information systems are used to formulate and execute business actions. The e-action level thus corresponds to the performance of an social action within the business context. The e-interaction level is required in order to analyse how ae-messages (and thus e-actions) are formulated (and thus decomposed). The e-interaction level is the ‘atomic’ level of action that is of interest without loosing the business focus. It would, for purposes other than the evaluation of actability, certainly be possible to decompose human-IS interactions into even more detailed levels (cf. Shneiderman, 1998).

At the other end of the scale, the concept of use-situation is used as an analytic delineation of meaningful sets of e-actions (with corresponding e-interactions). As pointed out in Section 4.5.3, the reason for this choice is the way human-IS interactions and also automatic operations thereby become integrated in the business model and the business context becomes emphasized. The important thing is that the concept of use-situation provides a theoretically justified and highly practical way of delineating appropriate analysis units.

Another candidate for high-level delineation of IS-related action would be that of the activity. In VIBA, and particularly when working with Action Diagrams, the activity is used as a high-level aggregate of e-actions. These aggregates are then grouped, or maybe even decomposed, into use-situations. The concept of activity is reserved for this more general purpose, in which activities may actually correspond to use-situations, e-actions, or even e-interactions, depending on the level of granularity required to understand the business process being modelled. In general, use-situations
and activities provide different means for grouping e-actions emphasizing either the process structure (activities) or system use (use-situations). From the perspective of IS design, focusing on IS use while keeping the link to the process structure is straightforward.

### 13.2.4 Actability and the Suggested Evaluation Process Model

The relation between the suggested evaluation process model and the concept of actability is not as straightforward as the relation between the analytical tools and the concept, as discussed above. The reason is that actability represents a way of conceiving information systems, and says nothing explicit about how to work with evaluation. However, the very existence of a theoretical concept as a foundation for what to evaluate implies that evaluation is guided by the theory and thus that some quality ideals inherent in the theory are used to direct attention during evaluation. Therefore, a method for actability evaluation will inherently contain elements of ideal typical evaluation.

Note that such evaluation does not necessarily have to correspond to heuristic evaluation, in the sense of Nielsen’s approach to usability inspection (Nielsen, 1994). In our work on ideal typical actability evaluation we have chosen to use an approach similar to Nielsen’s (1994) but with two major differences. First, the set of heuristics differ and thus partly the scope of evaluation. Instead of his ten usability heuristics (Nielsen, 1994) we have used the nine actability principles (see Section 12.3.1). Second, we acknowledge business and individual goals to play a part in the evaluation, allowing for the heuristics to be situated. This second aspect is not as clear-cut as the first, and may be regarded a matter of nuances. However, it seems that Nielsen’s (1994) approach assumes that an evaluation works with a set of already known and well-defined goals, which is not necessarily the case with actability evaluation. Actability takes the intentional (social) actions performed within a business context and therefore intentions and goal structures are important to critically examine and evaluate in relation to the IS used for achieving the goals.

A further epistemological standpoint inherent in actability evaluation is that information systems must be evaluated in-context. This is in line with other recent approaches to a more contextual understanding of the use of computers and IT in the workplace (Beyer and Holtzblatt, 1998; Kuutti, 1999; Bødker and Buur, 2002). This is the reason to favour situational evaluation in which actual users actual use of the IS is the foundation for understanding.

The suggested actability evaluation process model combines these two knowledge elicitation approaches: ideal typical and situational. These two approaches have been expressed as two distinct phases in the method. In practice, as we shall see below (Section 13.3.1), these two are highly intertwined and may very well be applied alternately during evaluation since they enrich each other.
13.3 Empirical Actability Evaluation Lessons Learned

As explained in Section 11.6, the empirical work on actability evaluation has resulted in ideas and results (see Figure 8-3, p. 137) that have continuously been incorporated in the method and in the concept of actability per se. These ideas thus concern the approach level of interest (see Section 11.3). In addition, a lot of practice level data has been collected following the method. This section presents reflections on the approach level, and uses insights gained at the practice level as rationale for the method and the concept of actability. The presentation thus focuses on the evaluation process model and the analytical tools used. Throughout the text, the labels given to the different sources of data as described in Table 11-3, p. 205 (the What and Source columns), are used. Note that quotations are the author’s translation from Swedish.

13.3.1 Applying the Evaluation Process Model

The main finding related to the use of the suggested evaluation process model relates to the separation of ideal typical and situational analysis into two distinct phases. During Application 5, this worked well and a ‘waterfall’ process was used where phenomena identified during ideal typical analysis were brought to situational analysis and used as a guide to direct attention, alongside a systematic search for other phenomena not easily identifiable without the actual users’ participation.

The separation into two phases worked well probably because the Internet Bank was quite intuitive in terms of possible actions to perform. In addition, one of the evaluators had several years of experience of working with the system. Since the evaluation was performed without cooperation with the system host (the Bank), there was no possibility to ask questions. Since this fact was known from the outset, the evaluators chose not to invoke users (in terms of system owners as communicators and interpreters) at this stage, simply because it was not possible. During Application 7, the situation changed a bit. Parts of the system were hard to understand for an evaluator that was unfamiliar with the system and its business context. Therefore, there was a need to contact users early in the process to clarify. This way the ideal typical phase and the situational phase were more or less blended into an integrated process of understanding and evaluating the Intranet.

The lesson learned is that it is hard to be too fundamentalist in separating ideal typical from situational evaluation. The question is if it is at all desirable? To learn a new system, and the business goals framing its use, at least managers must be consulted early in the process. The experience from Application 5 is that the two phases should be regarded as ‘ideal types’ (in a Weberian sense) and integrated during evaluation as required by the specific situation.

Note that Application 6 did not make use of the distinction between the phases at all but focused entirely on the ideal typical evaluation by an evaluator with much experience of the evaluated system.


60 In a sense this meant that there actually was a user involved also in the ideal typical evaluation.
A further reflection concerns the situational analysis in Application 5, during which it felt as if we were disturbing the users by asking questions about all IS functions and actions performed at the interface. This is probably because we evaluated on the detailed level of e-interactions. Another problem was that users performed their tasks quickly and it was sometimes hard to query how different functions were perceived. Perhaps this is inherent in the contextual enquiry approach, which implies that you study the users’ actual work.

During Application 8, a situational analysis only evaluation was performed. The evaluators choose to strictly follow Nielsen’s (1994) heuristic evaluation but used the actability principles instead of Nielsen’s heuristics to facilitate comparison between the two approaches. On reflection, during I8 one of the evaluators (R8) suggested that this might not have been an optimal approach. Since Nielsen’s (1994) heuristic evaluation focuses on the graphical user interface while the actability heuristics focus more on the relation between the system and the business context, a more thorough understanding of the business was required. This indicates the need for an initial survey and creation of process and task descriptions, as suggested by the actability evaluation process model. R9 (the other evaluator) confirmed R8’s reflection during I9.

### 13.3.2 Using the Actability Principles

The actability principles presented in Section 12.3.1 have helped to emphasize important parts of the evaluated information systems, from an actability perspective. During ideal typical analysis of the Internet Bank and the Intranet (Applications 5 and 7), several potential actability problems were identified. These problems were followed up during the corresponding situational analyses. We could then check if the users perceived the potential problems that had previously been identified.

Let us illustrate with an example from Application 7 (the Intranet) that has to do with visible actors and action responsibility.

#### 13.3.2.1 Using The Actability Principles in the Intranet Case

When booking resources through the Intranet a specific screen document is used to show information about a particular booking (a specific resource at a specific time interval). Among other things, the document displays the names of two people who are responsible for the resource, hereafter referred to as resource owners. These people are the ones who approve bookings of the particular resource, and at the bottom of the screen document one can read one of the following three phrases:

- (a) Status: This booking awaits an approval from the person responsible for the resource [the resource owner].
- (b) Status: This booking is approved by the person responsible for the resource [the resource owner].
- (c) Status: This booking has been denied by the person responsible for the resource [the resource owner].

Furthermore, the document displays the name of the person who has made the booking. At first this seemed excellent. The communicator of a booking request is identified. Two possible communicators of an approval of that booking are also shown, together with an explicit recognition of the acts performed: a booking request and a corresponding approval or denial. It turned out, however, that some resources do not
need ‘manual’ approval of bookings. Instead someone can book a resource directly, and get it approved automatically. In these cases, the approval is delegated to the Intranet. The problem is that it is not the resource owner that makes such delegations effective. Instead, there is a further actor involved, the one who manages resources and appoints ‘responsible’ people, hereafter referred to as a resource manager. This can be done without even notifying the resource owner being appointed. There are manual routines to secure that resource owners are informed about their responsibilities, but there is no support in the system for this. This may lead to situations in which a resource owner does not know that people making bookings regard him or her as the one making promises about the availability of resources. This was also a concern expressed by a resource manager who suggested that the person ‘behind the responsible person’ actually should be identified together with the ‘responsible’.

During an interview with a person acting as resource owner it became clear that he was actually unaware as to whether he was responsible or not for any resources at all at that time. This was due to reorganization making another person at the department responsible for all the department’s resources. Nonetheless, it is remarkable that the previous resource owner did not know, and had not been notified about the status of the reorganization.

An interesting issue in relation to this is that the reorganization was being made because the previous resource owner, who is the one responsible for the physical resources, felt that it was too demanding to handle all requests related to resource booking. Sometimes the required acknowledgment was not possible to make in due time. The system clearly states that a response from the resource owner will be received within five days. At times, the (former) resource owner did not get the request in due time since he was frequently out of the office, and hence the acknowledgement was not considered very important. To quote the (former) resource owner:

‘If a request is pending, no one requests that resource anyway. And if the resource is still [physically] available [at the time it is needed], they [the requesting booker] will use it anyway.’

Therefore, the responsibility was being appointed to another person working in the reception at the department. Of course, such delegation is fully acceptable and understandable. Still, it is questionable if the term ‘responsible’ is appropriate to use in the system, when in fact, it is more relevant to speak of a ‘booking handler’ or something similar.

Furthermore, if a resource owner quits, that person may still be ‘appointed’ as responsible for a resource in the system. The cancellation of such appointments is made through manual lookup and editing by the resource manager, who must know what resources the former resource owner was responsible for. There are no search facilities to find the resources of a particular resource owner. This condition was considered to be a ‘weak link’ by the resource manager.

Another related problem identified during the ideal typical evaluation of the Intranet was the fact that if a resource is deleted in the system altogether, it may still have bookings pending. That is, there is no consistency check in the system that bookings exist before a resource is deleted. If such a deletion is made and someone wants to check his or her bookings, the following highly uninformative error message is displayed:
ADODB.Field error '800a0bcd'

Either BOF or EOF is True, or the current record has been deleted. Requested operation requires a current record.

/Phone/Phone_Resource_Show.asp, line 30

The resource manager, who seemed to be disturbed by the fact, was not aware of this situation.

Altogether, this example shows how the actability principles help direct attention to phenomena not easily revealed at first. The principle of action elementariness made us focus on actor visibility. At first it was a pleasure to see that the people responsible (as communicators) were visible and easy to get in contact with (via links to contact information as well as direct e-mail links). Later on, the principle of delegated action made us investigate more thoroughly the actual responsibilities claimed by the system. This made us understand that the seemingly well-presented action responsibilities were not clear at all, but rather an important issue for future redesign of the system and its business context. It is, for example, necessary to sort out responsibilities and implement the required consistency and integrity checks in the system. The latter issue is also closely related to the principle of recorded action – the action of reporting the deletion of resources was missing in the system. Furthermore, the principle of action potentiality made us focus on the information needed in performing different tasks, which revealed the problem of lack of search facilities regarding the allocation of individuals responsible for resources. This proved to be a problem for both the resource manager as well as the former resource owner. It was also interesting to see how the principles of delayed interpretation and delayed feedback helped to direct attention to the promise made in the system regarding the time frame of response to a booking request. This was understood to be important information but it was suspected that the promise was not always kept. This turned out to be true, and was also a major concern for the resource owner. The solution to the problem was to delegate the responsibility. However, the principle of remote interpretation made us suggest that requests could be handled via mobile devices when out of office. This turned out to be something that the former resource owner found appealing and he even suggested it as a redesign proposal to be considered (which also included the integration of the Intranet with personal PDA organizers).

13.3.2.2 Some Further Reflections on the Use of the Actability Principles

As stated above, the actability principles evolved during Applications 5 and 6. During evaluation of the Internet Bank (Application 5), a new actability heuristic was formulated: *good conditions for action in required information*, which was later incorporated in the principle of action potentiality. The lack of such a heuristic was identified when we analysed the result from the user tests. We did not have a heuristic that explained the identified phenomenon that several users were dissatisfied with having to input information that they did not consider meaningful. An example is that dates could not be provided to the system the expected way using hyphens, i.e. as in 2002-01-25, which is the common Swedish way of writing dates.

Three additional heuristics were identified during the evaluation of the Booking System in Application 6. These were called *accurate timing*, *interpretation initiative* and *distribution of actions*. They can be illustrated by the example of a lecturer who –
due to sudden illness – wants to make a last minute change in the schedule. The system supports the re-scheduling. However, there are several things left to do to ensure that his students receive this important information. This is especially important, since many of the students commute to the university. Some problems can only be solved by a number of manual actions. In this case, the student has to take the interpretation initiative. Most students print their schedules at the start of the course, and then check the IS for changes periodically. However, there is no guarantee that they will be aware of changes (until it is too late). If e-mails were sent automatically to the course participants whenever a last minute change in the schedule is made, it would be more likely that students would receive the information in due time. If even more advanced technology was utilized, the message could be pushed to the students (for example, using SMS messaging). In this case, it is obvious that the accurate timing heuristic is closely related to the interpretation initiative. These heuristics are now considered part of the principles of delayed interpretation (accurate timing), remote interpretation (interpretation initiative), and delegated action (distribution of actions).

On reflection, it is not straightforward to study an isolated IS within an organization. In the example above, the teacher could obtain a list of e-mail addresses of the students from another system (which had to be done manually) and use it to notify all the students. Ideally, these actions would be performed by the booking system. The principle of delegated action helps to focus on this type of problem.

The question of delimitation is discussed by Kindgren and Messo (2003) and was also brought up by R8 and R9 during I8 and I9. During Application 8, the evaluators chose to delimit the evaluation to a small part of the IS under scrutiny. Such delimitation made it hard to apply all actability principles, which suggest that the interpretation of communicated messages is also important to consider, and this interpretation may be done in another part of the system. ‘Is it at all possible to evaluate only a part of a system? Doesn’t that oppose the whole idea?’ (R8 during I9). Obviously, it must be possible to delimit a study; a study must have a focus. It is important, though, to consider the actability principles already during this delimitation so that important (business critical) considerations are not overlooked. This approach is in line with the contextual approach to process descriptions as discussed in Chapter 9, and is also a reason for suggesting the use of Action Diagrams to describe relevant business processes during actability evaluation.

A further note is that it takes a great deal of time to evaluate a system taking all the design implications of the actability principles into consideration (cf. Nielsen, 1993). This was experienced during all four applications. Consequently, there is a need for support to identify what to focus on during the evaluation.

One interesting point that was raised during Application 8 was that both evaluators (R8 and R9) considered the actability principles more difficult to apply than Nielsen’s (1994) heuristics. Their conclusion was that the actability principles require a more thorough internalization of the underlying theory compared to Nielsen’s heuristics. ‘The way the ideal typical phase of actability evaluation is described to date means that we do not see that it is possible to make an evaluation in a few hours… Actability evaluation demands more of the evaluator concerning … knowledge of the concepts and values of the underlying perspective, and in our view also of commitment’ (Kindgren and Messo, 2003, p. 49) (author’s translation from Swedish). This seems to contradict the idea of an inspection method (such as heuristic evaluation) since such methods are supposed to be used quickly and efficiently without too much pre-understanding. This problem may be approached by providing better hands-on guide-
lines, but may be difficult to completely overcome due to the quite heavy theoretical underpinning of actability.

13.3.3 Using the D.EU.PS. Model

The D.EU.PS. model was used explicitly in the situational analysis of the Internet Bank and the Intranet. The enquiries were designed as typical questions based on the actability principles, the D.EU.PS. model and problems that were identified during ideal typical evaluation. We asked questions such as: Is this functionality desired? Is it satisfactory? Is there anything that you lack in the system – something you would like to do that the system does not allow for? As an example of the use of the D.EU.PS. model, let us turn to how it was used in Application 7, the evaluation of the Intranet.

13.3.3.1 Using the D.EU.PS. Model in the Intranet Case

As described in Chapter 11, Application 7 was conducted on an intranet (the Intranet) mainly used for dissemination of information and booking of different resources (rooms, projectors, et cetera). The focus in this example is on the booking of resources. This is a part of the Intranet where users can, for example, book a room at another department in another city. As described in Section 13.3.2.1, each resource has one or several resource owners who are responsible for the resource. You can, for example, contact a resource owner if you have any questions about a particular resource, for example, how far a particular room is from the coffee machine, which the Intranet usually not tells.

The way the suggested evaluation process model was used during Application 7 is described in Table 13-1 in terms of four steps (see also Section 13.3.1). The first step concerns ideal typical analysis and Steps 2, 3, and 4 concern situational analysis.

<table>
<thead>
<tr>
<th>What to do?</th>
<th>How?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1: Identify existing functionality; what you can do with the IS.</td>
<td>Documentation analysis, studying the IS and interview system owner.</td>
</tr>
<tr>
<td>Step 2: Design an evaluation protocol.</td>
<td>The evaluator relates the identified (existing) functionality to the five overall D.EU.PS. categories.</td>
</tr>
<tr>
<td>Step 3: Identify phenomena (Evaluation).</td>
<td>Observe and interview users. Classify observations with respect to the D.EU.PS classes.</td>
</tr>
<tr>
<td>Step 4: Analysis.</td>
<td>Relate observations to causes and effects. Problem relations, Strengths relations: Causes and effects.</td>
</tr>
</tbody>
</table>

In the remainder of this section, the four steps of Table 13-1 are discussed in detail together with reflections regarding the applicability of the D.EU.PS. model.

Identifying Existing Functionality

Trivial as it may seem, the task of identifying the existing functionality raised questions such as: What characterizes existing functionality? Is it the designer’s intended functionality that constitutes the existing functionality? If users use a function in a different way than what was intended, does the original function still exist? How do you determine the existing functionality? Can the objective action space of designers and users change with knowledge of how the information system is used?

When an evaluator seeks to understand an IS, the existing functionality corresponds to that functionality which the evaluator understands to be in the system – what the evaluator believes users can do with the system. That is, the evaluator’s understanding of the system is what defines the existing functionality, and the evaluator thus has the authority to decide on what exists and not. Such understanding may be reached through eliciting designers’ intentions and studying the IS and its documentation. The evaluator’s authority is thus based on an interest in creating a shared understanding about the system and its use. The class of existing functionality typically extends over time as the evaluator learns more about the system and its use. Perceived functionality, on the other hand, is what the user believes exist, the user’s understanding. Both perceived and existing functionality may differ from what the designer intended, i.e. from the functionality the designer believes exist. Particular properties of an IS that are regarded as an existing function may be used in a completely different way than was intended, and thus constitute a further perceived but non-existing function. As discussed in Chapter 12, the goal of a design should be to create a shared understanding of the IS so that intended, perceived and existing functionality coincide; that designers, users and evaluators agree on what the system can do. (Of course, sometimes a designer may also act as evaluator).

The distinction between existing and perceived functionality makes it possible to identify existing functionality initially, without involving actual users. In the Intranet case, this identification was achieved by studying the purpose of the artefact and the functionality it delivered. The functionality is the benefit of the artefact, what you should be able to do in a specific context (Holmlid, 2002). An IS implements certain functions. These functions together constitute the action potential of the IS – what one can possibly do with it. The action potential of the IS, in turn, spans an action space for the user to act within.

How, then, do you go about determining the existing functionality? According to Löwgren & Stolterman (1998), functions tend to be expressed with two words: a verb followed by a noun, representing an action and an object acted upon, for example, ‘register customer’. A function thus corresponds to an action that the IS supports. The functionality of an IS goes hand-in-hand with its form. As discussed in Chapter 4, functions are made visible and actable through screen documents – screen documents are what give functions a form. A screen document can be seen as multifunctional in the way that it can support the conduct of an action (informing), it can function as an action media in the execution of an action, and it can contain results of action for users to interpret (Cronholm and Goldkuhl, 2002).

In the case study several supported user actions with corresponding IS actions (functionality) were identified in the Intranet. These are described in Table 13-2 together with the business roles of the actors performing and interpreting the actions. A booker is someone that can perform bookings in the system, and a resource owner is someone appointed as responsible for a particular resource (see above). Since all re-
source owners can also make bookings, a resource owner can also perform actions described as performed by bookers in Table 13-2.

**Table 13-2**: Example of supported user actions and corresponding IS actions (functionality) found in the Intranet. (adapted from Eliason and Ågerfalk, 2003)

<table>
<thead>
<tr>
<th>User Action</th>
<th>Performer</th>
<th>Intended Interpreter</th>
<th>IS Action (System Function)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search resource</td>
<td>Booker</td>
<td>Booker</td>
<td>Search for and display the name of a resource, based on different criteria</td>
</tr>
<tr>
<td>Show information about booking</td>
<td>Booker</td>
<td>Booker</td>
<td>Display information about a particular booking (a particular resource at a particular time)</td>
</tr>
<tr>
<td>Request resource</td>
<td>Booker</td>
<td>Resource owner (and other bookers)</td>
<td>Mark the resource as ‘request pending’ for the requested time and notify resource owner</td>
</tr>
<tr>
<td>Cancel booking</td>
<td>Booker</td>
<td>Resource owner (and other bookers)</td>
<td>Mark the resource as ‘available’ and notify resource owner</td>
</tr>
<tr>
<td>Respond to request (acceptance or denial)</td>
<td>Resource owner</td>
<td>Booker (all bookers, and specifically the requesting one)</td>
<td>Mark the resource as ‘booked’ or ‘available’ depending on decision and notify requesting Booker</td>
</tr>
<tr>
<td>Recall accepted request</td>
<td>Resource owner</td>
<td>Booker (all bookers, and specifically the requesting one)</td>
<td>Mark the resource as ‘available’ and notify requesting Booker</td>
</tr>
</tbody>
</table>

Identifying existing functionality makes it possible to discover situations where users use a function differently than what was intended by the designer. Knowledge of how the information system is used leads to the designer changing his perception of the user’s action space. This, in turn, may lead to the designer changing his subjective understanding of the IS as well, conforming to the user’s view, or that the designer tries to change the user’s perception of the function. This way, what is characterized as existing functionality and perceived functionality is dynamic and contingent upon users’ and designers’ experiences of the IS in use. Of course, such dynamics cannot be identified without involving users.

**Designing Evaluation Protocol**

When the existing functionality had been identified, a protocol was created in which it was possible to take notes about phenomena and relate them to the five different D.EU.PS. categories (see Table 12-5, p. 226). This way, the protocols used during evaluation were explicitly based on the D.EU.PS. model. The protocol supported us (R1 and R6) in directing attention to key aspects and reminded us to ask questions such as: Is this functionality desired? Is it satisfactory? Is there anything you lack in the system? An example of the use of the protocol is given below (Table 13-4).

**Identifying Phenomena**

The evaluation performed in the case study included both observations and interviews. That way the evaluation could be extended to include not only what was actually in the system, but also to what users perceived the system as providing and what they thought the system should provide. Thus we could work closely with users to learn both the users’ tasks and the ways they used the IS. Based on Beyer and Holtzblatt (1998) this...
approach tries to resemble a ‘new-on-the-job’ situation between user and evaluator (see Chapter 12).

Berglind (1990) stresses that it is through talking with people about what they want and do not want, can and cannot do, and what restricts them that you can get an understanding of their experience. During evaluation, the goal is to enable users to show and describe how their actions are performed, and to express reasons for action within their actual work context. That is, to express their knowledge in action (Schön, 1983). The suggested approach (see Chapter 12), which is basically a combination of thinking aloud and observation, was favoured because it enabled users, at least to some extent, to express tacit knowledge (Polanyi, 1983). Together with expressed thoughts and observations, we had the opportunity to understand what was problematic or unproblematic (which could be observed from user actions) and to learn why and how it caused effects. Data collection was thus based primarily on positive and negative aspects encountered in the work practice. This way different phenomena (problems and strengths observed in relation to the Intranet) and their causes and effects could be identified. Table 13-3 summarizes this third step in the evaluation: Identify Phenomena. Table 13-4 shows an example of a document from this step.

Table 13-3: Summary of Step 3: Identify Phenomena. (Eliason and Ågerfalk, 2003)

<table>
<thead>
<tr>
<th>Objectives</th>
<th>What to Study?</th>
<th>How?</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identify phenomena (problems and strengths) their causes and effects.</td>
<td>IS in use</td>
<td>Data collection as a combination of observations and interviews.</td>
<td>Protocol (see Table 13-4).</td>
</tr>
<tr>
<td>Understand which class a problem or strength is related to, which gives a basis for assessment and, possibly, change proposals.</td>
<td>IS in use</td>
<td>Classification of observed phenomena in accordance with the D.EU.PS. model.</td>
<td>Protocol (see Table 13-4)</td>
</tr>
</tbody>
</table>

Table 13-4: Example protocol for an observed phenomenon related to the function Recall Accepted Request in the Intranet.

<table>
<thead>
<tr>
<th>Function</th>
<th>Desired</th>
<th>Existing</th>
<th>Utilized</th>
<th>Perceived</th>
<th>Satisfactory</th>
<th>Description of Phenomenon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recall accepted request</td>
<td>D</td>
<td>E</td>
<td>U</td>
<td>P</td>
<td>¬S</td>
<td>‘I believe that you should call and check with the person who has booked the resource… Perhaps they have booked a lot of people on a course… You want to know why your reservation was recalled not only that it was recalled.’ (A resource owner during Application 7, author’s translation from Swedish)</td>
</tr>
</tbody>
</table>
The phenomenon observed (described in Table 13-4) was that a resource manager did not find the function Recall Accepted Request satisfactory because the function did not provide an opportunity to explain why he recalled accepted requests. The function, which existed in the system, was desired and used but not with satisfaction. In this particular case, the D.EU.PS model thus helped us to pinpoint how the function was used and why. The function as such was desired but it did not give the user the required action space: to give an explanation to the booker who had been turned down.

**Analysis of Causes and Effects**

Throughout the evaluation, identified strengths and problems were analysed in terms of cause and effect and classified based on the classes of the D.EU.PS. model. This classification made it possible, for example, to discuss who would be affected by a problem and the importance of correcting it. By classifying phenomena as strengths and problems, and relating them to each other, a better understanding of the effects of the IS can be achieved. This is in line with the analysis of problems in VIBA (see Section 9.5.1.2) and results in directed graphs of problems and strengths showing which problems cause other problems, and which strengths contribute to other strengths. This way, the most important problems to correct and the most significant benefits could be pinpointed. Table 13-5 summarizes this final step in the evaluation.

**Table 13-5: Analysis of Causes and Effects.**

<table>
<thead>
<tr>
<th>Objective</th>
<th>What to study?</th>
<th>How?</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identify relations between strengths</td>
<td>Documentation from the evaluation conducted so</td>
<td>Data analysis:</td>
<td>Function summaries</td>
</tr>
<tr>
<td>and problems.</td>
<td>far</td>
<td>Relation of observation to causes and effects</td>
<td>(see Table 13-6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Relationships between problems and</td>
<td>Problem Graphs and Strength Graphs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>between strengths; causes and effects.</td>
<td></td>
</tr>
</tbody>
</table>

The D.EU.PS. model made it possible to discuss the functionality of a system in terms of what is desired and what is not, what exists and what is missing, what is actually utilized and what is unnecessary, what is believed to exist, and what can be used with satisfaction? This in turn made it possible to identify and discuss different users’ perceptions of and attitudes towards a system, and how these perceptions and attitudes changed over time. This analysis was summarized as shown in Table 13-6.

From the example in Table 13-6 we can see that one phenomenon observed was that a resource owner (User 2) tended not to cancel bookings even though the resource was no longer needed. The reason was that the resource owner was not aware she could; she did not perceive the function. This can be due to the fact that User 2 was recently promoted from booker to resource owner. The direct effect was that the room remained marked as booked in the IS when it in fact was available for booking. User 1, on the other hand, knew about the function but used it unwillingly due to the function’s limitation in creating the required action space. The resource owner (User 1) considered that the IS did not support him to communicate the reason for the recall of the booking to the affected booker.
Table 13-6: The Intranet function Recall Accepted Request (quotations are author’s translation from Swedish).

<table>
<thead>
<tr>
<th>Function</th>
<th>User(s)</th>
<th>Class</th>
<th>Cause</th>
<th>Phenomenon</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recall Accepted Request</td>
<td>User 1 (Resource owner)</td>
<td>D U P¬S</td>
<td>The possibility to give information about why the reservation is recalled is missing in the information system.</td>
<td>‘I believe that you should call and check with the person who has booked the resource,… Perhaps they have booked a lot of people on a course… You want to know why your reservation was recalled not only that it was recalled.’</td>
<td>The function is used, but only after a phone call to the booker has been made. This is because the IS function does not permit noting why a reservation has been recalled.</td>
</tr>
<tr>
<td></td>
<td>User 2 (Resource owner)</td>
<td>D E¬U ¬P</td>
<td>The resource owner is not aware of the possibility of recalling a request once accepted.</td>
<td>‘I cannot recall a reservation, even if I know that the room is empty, like if the meeting has been cancelled.’</td>
<td>The room is marked as booked in the Intranet, when it in fact is available.</td>
</tr>
</tbody>
</table>

Some Reflections on Application 7

Prates et al. (2000) argue that a system can be perceived as a ‘discourse deputy’ for the designer. This means that the IS as deputy communicates to the users what the designer has predicted and that users can usually only communicate with the designer’s deputy, not with the designer. Therefore effects of information systems design can only be identified in the users’ perception of the information system, in terms of the D.EU.PS. model. Effects also exist as the consequences the design has for user’s actions based on that perception. For example, User 1 (see Table 13-6) regards the function Recall Accepted Request as unsatisfactory and therefore decides to take action outside of the system by making a phone call before using it. This effect may eventually lead to unintended business effects such as that the resource booking as a whole is not used (if, for example, the resource owner cannot get hold of the booker he tries to call).

A consequence (a non-intended effect) of the non-action of User 2 (see Table 13-6), i.e. to not recall accepted requests, is that other users of the system believe the room to be booked. Thus, effects arise in system usage that in turn can have serious effects on business effectiveness and efficiency. This is truly an argument for the importance of viewing information systems as mediating tools for business action and communication between different actors.

13.3.3.2 Some Further Reflections on the Use of the D.EU.PS. Model

As described in Section 12.4, protocols were used during the analysis based on the D.EU.PS. model. The model, as an analytic tool, supported the evaluator in the situational evaluation to direct attention to key aspects. For example, we discovered that in order for a user to perceive a function as desired, the benefits of the function must be
communicated so that users understand why to use the functionality (the action potential provided by the system must be comprehensible). The perception of what is considered desired in the system can also differ between users and management. For example, security codes have to be provided to the Internet Bank several times during the bill-paying process. This was considered meaningless by the bill-paying users – it required more time, but it did not make the system feel more secure (¬D U ¬S, that is). On the other hand, the Internet Bank provider probably wants to promote a feeling of security and prevent misuse. A further notice is that to be able to utilize a function, a fundamental aspect in the system is that the user knows how to use it. For example, an Internet Bank user did not know how to change already registered information and therefore did not use that functionality (D E ¬U ¬P, that is). When the users said that a function was unsatisfactory, the terms ‘sure’ and ‘unsure’ were frequently used descriptions. For example, a user of the Internet Bank felt unsure about what occurred (¬S, that is). A statement that also reappeared was that functionality was perceived as ‘bad’ (¬S) or ‘good’ (S).

13.3.4 Using the Layered Model of Action

When applying the evaluation approach, a choice is given regarding the level of granularity on which to perform the evaluation. Based on the concept of actability we may distinguish between three levels: the use-situation, the e-action, and the e-interaction performed to formulate ae-messages and execute e-actions.

After evaluation of the Internet Bank in Application 5, it became obvious that, despite the rather narrow evaluation, we had produced a significant amount of documentation. The problem was that we had pursued the evaluation on the e-interaction level. By choosing a more aggregated level, it is possible to make an overall judgement of a larger part of a system without digging into all the details. The lesson learned was that there is a need for support in the evaluation model to choose the level of evaluation.

13.3.5 Repercussion on Actability

The evaluation of the Intranet in Application 7 led to a refinement of the ae-message concept into that presented in Section 5.5.2. Up until then the ae-message concept and its relation to screen documents had not been analysed in the same detail. During that application, it became clear that in order to understand in detail the message processing in a system, there is a need to explicitly define output messages and, by way of precedence analysis, derive required input messages and match those with the actions performed at the user interface. Up until then, we had worked under the assumption that what is shown in a screen document is a collection of input ae-messages.

The work on actability evaluation also made us gradually more and more explicit about the concept of IS use as social action. Although that notion had underpinned the work from the outset, we had not considered all of its consequences. It was during Application 6, when the heuristics related to remote interpretation and interpretation initiative was formulated that we stepped completely away from focusing on one user in one use-situation at a time. Even though intended interpreters had always been important, it broadened our scope to follow up the consequences which arise if user’s intentions do not become effective in due time.
Part V

Actability Studies
Chapter 14

Study I: The Case of Internet-Based Systems


14.1 Chapter Outline

This chapter is an inquiry into the empirical grounding of actability. The chapter describes the structure and the application of an analytic framework based on actability and the semiotic framework (Stamper, 1996). The framework has been used as a tool to direct attention during a qualitative analysis of Internet-based (information) systems. The results show that actability and (a re-interpretation of) the semiotic framework can be used effectively to gain understanding of specific information systems phenomena.

The chapter is structured as follows. After giving some background for the study, the approach used for the empirical grounding of actability and its relation to internal and external theoretical grounding is elaborated further. Secondly, the operationalization of actability and re-interpretation of the semiotic framework into a descriptive analytic framework is presented. Thirdly, results from the analysis of Internet-based systems performed using the analytic framework are presented. Finally, the study concludes by reflecting upon the analysis performed and the explanatory power of actability.

14.2 Background of Study I

In a sense, actability can be understood as summarizing a theory of information systems as information action systems – an action theory of information systems. One characteristic of the theory is that it is concerned with human action in relation to information systems and that human action also is its ultimate purpose. It represents knowledge about action, intended for action.

The goal of this work is to strengthen the empirical grounding of actability with a special focus on an analytic framework for analysing information systems phenomena according to actability. More specifically, actability is used to gain a better understanding of Internet-based information systems. This chapter aims to provide concrete ex-
amples of actability concepts and to establish that these concepts are useful to discuss this type of system. That is, to present good, empirically justified reasons as arguments for the concept of actability.

This work was performed in co-operation with an industrial partner, a large multinational software development company. This partner’s interest in this work is to gain a better understanding of Internet-based systems in order to tailor their development process according to its needs. This interest stems from the fact that during the last decade, we have seen a shift in focus of information systems development (ISD) from ‘traditional’ information systems to distributed applications delivered via the Internet – Internet-based information systems. This has led to a shift in the technology used to provide software solutions. Even the Internet as a platform for applications has changed. In the early days of the Web, the platform was mainly used for e-mail exchange and trivial websites. Although these services still constitute a great deal of the Internet, more sophisticated applications are increasing in number. They are sophisticated both in content and in the technology on which they are based. Mobile Internet is, for example, the latest of trends that is making information systems available to users. Consequently, this study is not restricted to Web-based systems, as, for example, Isakowitz et al. (1998) are.

These more complex artefacts impact on organizations to a greater extent than before. The expansion in scope of these artefacts also presents new opportunities for doing business. Many hypotheses have been made about how these artefacts will change the entire structure of industries (Hoffman et al., 1997; Tenenbaum, 1998). Nonetheless, there is a lack of understanding of the development of this type of system. A reasonable question to ask is how this type of system and its development differs from traditional information systems. Practitioners seem to emphasize the newness of technologies, design, the development environment and the development process. For example, one of the projects we participated in aimed at developing an Internet-based online ordering service. The project team had to manage a number of challenges, including new technology (Windows DNA), a new development process (Rational Unified Process), and a new type of deployment (online). In their eyes, Internet-based systems were a new and mysterious concept.

In the literature, there seems to be a focus on Web applications, or more specifically on websites, as an end product. For example, the usability aspect of websites has been discussed frequently (e.g. Nielsen, 1999). Other authors address the modelling aspect of Web-based information systems (Takahashi and Lang, 1997; Conallen, 2000) and the technical aspects (Spreitzer and Janssen, 2000; Urien, 2000; Juric et al., 2000). Overmyer (2000), focusing on website requirements engineering, has applied a somewhat broader perspective, and, complex websites excluded, found three major differences: different focus, different disciplinary emphases, and shorter life-cycles. Still, the literature gives only a fragmented description of Internet-based systems. With only a fragmented picture of the system, it is difficult to see the effects of changes in different parts of a project or its conditions, and predicting these effects is even harder.

With the increasing size and complexity of Internet-based systems development projects it is important to gain a comprehensive picture of this type of system and its characteristics. This chapter focuses on remedying the fragmented picture of Internet-based systems by presenting an empirically justified description of its characteristics. In addition to get empirical experience from working with actability, this study aims, in contrast to, for example, Isakowitz et al. (1998), to demystify the Internet-based system and its development by pointing at similarities with rather than differences from tradi-
tional ISD, with special focus on user requirements. Today we can find games, e-commerce systems, corporate intranets, *et cetera*, based on the Internet platform. With such a wide variety of artefacts available, attention is delimited to software artefacts for business action and communication, in accordance with the overall aims and demarcations of this dissertation (see Chapter 1).

The choice of Internet-based systems as the topic for empirical grounding of actability is not self-evident. The reason for doing so is that Internet-based systems often imply that communication between human actors (typically between supplier and customer in a business setting) is to a large extent performed through the system. Therefore, the qualities promoted by actability seem particularly crucial in that type of system.

### 14.3 Actability Study I Research Method

As suggested in Chapter 3, actability can be regarded as an instance of what Goldkuhl (1999) refers to as *action knowledge*, i.e. ‘theories, strategies and methods governing people’s action in social practices’. Action knowledge might exist in different forms of abstraction – from ‘pure’ abstract theoretical knowledge, to knowledge directly applicable in everyday situations. In Chapter 3, it was argued that grounding of action knowledge (internally, externally and empirically) is to present such good reasons for it that other people will accept it as valid.

The result of empirical grounding can be focused on as a concept in its own right and be exposed to its own three-way grounding process, which implies that empirical grounding might also include elements of internal and external theoretical grounding. Based on these concepts Figure 14-1 shows the grounding of the analytic framework used in this study (cf. Table 3-3, p. 46). The principle is that the grounding of an operationalization (i.e. of a more concrete representation of a phenomenon), in this case the analytic framework, internally, externally and empirically yields an empirical grounding of the concept of which it is an operationalization, in this case actability.

<table>
<thead>
<tr>
<th>Concept</th>
<th>Internal Grounding</th>
<th>External Theoretical Grounding</th>
<th>Empirical Grounding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actability</td>
<td>Purely internal</td>
<td>Å ISD + RE</td>
<td>Å An actability analytic framework</td>
</tr>
<tr>
<td>Main Concept</td>
<td></td>
<td>Å CM (LAP + OS)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Å Usability</td>
<td></td>
</tr>
<tr>
<td>Description of Internet-based systems</td>
<td>Internal plus the analytic framework</td>
<td>Other descriptions of Internet-based systems</td>
<td>Method configuration together with industrial partner</td>
</tr>
</tbody>
</table>

**Figure 14-1:** Grounding of the concepts focused in this chapter (adapted from Ågerfalk *et al.*, 2002a).
In this work a qualitative research approach that combines deductive and inductive strategies has been adopted. This way of working is referred to as reflective research (see Chapter 3). The strategy adopted is comparable to what Walsham (1993) describes as the approach taken when a researcher uses theory as part of an iterative process to collect and interpret data. Figure 14-2 illustrates the way this strategy is used.

Figure 14-2: The research process (adapted from Ågerfalk et al., 2002a).

Figure 14-2 should be interpreted as follows. Actability and the semiotic framework (Stamper, 1996) have been (1) operationalized into (2) an analytic framework (see Section 14.4). This work also led to a deeper understanding of actability as such (3), and also served as (part of the) external theoretical grounding of actability. The analytic framework has been used (4) to direct attention during data collection (5) at Volvo IT. Collected data has then been abstracted (6) into categories describing various aspects of Internet-based systems (7) similar to Grounded Theory’s open coding and axial coding (Strauss and Corbin, 1998). During this abstraction, the analytic framework has also been used (8) as a tool to direct attention to relevant categories and phenomena. Additionally (9), categories have been related to, and inspired by, existing knowledge about Internet-based systems and their development. The emerging understanding of Internet-based systems has had continuous repercussions (10) on the operationalization process (and thus indirectly on actability itself, as well as on the analytic framework) and on the data collection.

14.4 An Analytic Framework Based on Actability

From the definition of actability (elaborated in Chapter 4) we can observe some central concepts for actability. Below, these concepts are addressed in turn to eventually arrive at an analytic framework that can be used to understand information systems phenomena according to actability – an operationalization of actability, as discussed above. Before doing that, the general information systems framework referred to as the semiotic framework (e.g. Stamper, 1996), will be presented in more detail than was done in Chapter 4 as it is used as an inspiration and external point of reference for the analytic framework.
14.4.1 The Semiotic Framework

Stamper (1996) introduced the semiotic framework as an attempt to create one single conceptual framework able to capture both the social and the technical aspects of information systems. The fundamental concept for the semiotic framework is that of the sign. Inspired by the classical definition by Peirce, Stamper (1996) defines ‘sign’ as ‘something which stands to somebody for something in some respect or capacity, in some community or social context’. Consequently, information is ‘carried’ by signs of different kinds (Stamper, 1996). Following the semiotic framework, signs (and hence information) can be studied at six different semiotic levels. That is, we can choose to focus on different aspects of signs ranging from their physical appearance to their social consequences. The six levels are referred to as physical world, empirics, syntactics, semantics, pragmatics and social world (see Figure 14-3).

Figure 14-3: The Semiotic Framework (Stamper, 1996).

The first three of these are referred to as the IT platform and the latter three as human information functions (Stamper, 1996). The IT platform can thus be thought of as a medium for the human information functions, and the same information functions can exist within different media. As pointed out in Chapter 4, within the human information functions, the distinction made between social world and pragmatics is problematic. Rather, these two levels are so intertwined that distinguishing them is probably not only conceptually inelegant, but also misleading. Stamper (2001) states that ‘Semiotics that excludes norms and attitudes as forms of information would be like physics with the concept of energy but without the concept of mass’ as an argument for adding the sixth level to the semiotic framework – the social world. This argument is sound in concept, but not in the way it is used within the semiotic framework. To understand why, and to see how the semiotic framework has been used in this work, we must first turn to the actability centre of gravity – the performance of actions.
14.4.2 Performance of Action Revisited

People perform actions to accomplish changes in the world – action is about making a difference. As discussed in Chapter 4, actions can be classified as material or communicative. Material actions aim at changing the physical state of the world. Communication actions (or speech acts) aim at changing socially constructed reality (Berger and Luckmann, 1967; Searle, 1996).

In this chapter the focus is on communication actions since they are the most important types of action in relation to the design of information systems. (See Chapter 4 and Goldkuhl (2001) for a more comprehensive treatment of both types of action.) Communication action means that a pragmatic action mode (illocutionary force) is attached to a semantic propositional content and formulated syntactically into an uttered sentence. The action mode represents what the speaker does in relation to potential listeners. Note that, as discussed in Chapter 5, actions are considered to be multi-functional and that an action might concern several illocutionary forces, one per function (illocutionary point). This is a pragmatic aspect of action. The propositional content represents what is talked about and consists of references to things in the world and properties predicated to those things. This is a semantic aspect of action.

Actions are performed through some material substrata – a medium. The medium is primarily an empirical and physical aspect of action. However, the medium might also affect the possible syntax of an action, and hence is partly a syntactic aspect. If, for example, the medium is ‘ink on paper’, the same action mode and propositional content may be communicated both as written text and as a picture. The medium can be thought of as the instrument used to perform action.

The effect of a communication action is a change in the social world, i.e. the establishment of a social fact (Searle, 1996). However, pragmatic intentions are usually directed from one actor to another and so they are social by nature. That is one reason to disagree with the terminology of the semiotic framework. Furthermore, the successful performance of a communication action usually requires knowledge of certain social facts established by previous communication actions. In general, any action requires certain external and internal prerequisites to be met (see Chapter 4). Therefore, we cannot study the pragmatic level of the semiotic framework without relying on the social world level, and so it seems that the whole idea of the ‘ladder structure’ is obstructed. Therefore it is not constructive to talk about the ‘social world’ and ‘pragmatics’ as different semiotic levels. Rather, it seems more appropriate to talk about pragmatics as the highest, and indeed a very social, level. Nonetheless, pragmatics must be understood in relation to existing subjective as well as inter-subjective knowledge. We can refer to this knowledge, which is both a prerequisite for and a result of action, as a cognitive base. The cognitive base can be divided into a subjective part and an inter-subjective part – a personal cognitive base and a shared cognitive base. The personal cognitive base consists of the person’s identity, personal values, norms and preferences, abilities, emotions, deliberations, intentions and plans, situational comprehension and attention, etc. The shared cognitive base constitutes the actors common ground (Clark, 1996) and consists of group belonging, shared values, norms and preferences, social facts, et cetera. (See Section 3.2.1 and Chapter 4, especially Figure 4-2, p. 57). Figure 14-4 depicts this actability reinterpretation of the semiotic framework with two actors communicating.

The performance of a communication action can be understood as consisting of the following seven steps, as the figure suggests:
1. The performing actor (the speaker) uses his personal cognitive base and matches it with the cognitive base shared with intended interpreters of the action (listeners) in order to formulate the communication action to be performed; that is, formulating a propositional content and embedding it into an appropriate action mode.

2. The action is expressed syntactically and performed through some medium.

3. Another actor receives the action, or rather, the message in which it resulted.

4. The receiving actor uses his/her personal cognitive base and matches it with the cognitive base shared with the speaker in order to interpret the performed communication action.

5. The receiver reacts cognitively in response to the interpreted action, which changes his/her subjective knowledge (personal cognitive base) as well as the shared cognitive base – a social fact has been established.

6. During action, an actor is continuously monitoring the world and reacting to changing circumstances. This conditioning and reflexivity is not something that happens after the action has been performed and interpreted, as the number six in the figure might suggest, but is a continuous process.

7. As a consequence of the conditioning and reflexivity, the cognitive base might change.

---

**Figure 14-4:** An actability reinterpretation of the semiotic framework (Ågerfalk et al., 2002a).

An important aspect of communication actions in general, and communicative action specifically, is that of *validity* (cf. the discussion on grounding, above). In order for a communication act to be considered as successful communicative action, an interpreter must accept it as valid. According to Habermas (1984), validity can be evaluated with respect to a set of universal validity claims. Based on his analysis, we can conclude that a receiving actor can value the information with respect to comprehensibility, truth, sincerity and (normative) rightness. That is, the information should be possible to comprehend syntactically, it should refer to the true (commonly believed) state of affairs (inter-subjective semantics), reflect sincere pragmatic intentions, and it
should be communicated in accordance with accepted social norms (inter-subjective cognitive base).

14.4.3 Information Systems: Software Artefacts for Human Action

Information systems should in the actability view be understood as software artefacts intended for human action. IT-based information systems are today obvious parts of businesses and should be acknowledged as important tools for business action and communication. What should be clear from the previous parts of this dissertation is that communication through an IS is an example of communication action where an actor communicates a message to another (or possibly the same) actor. In such communication, the software artefact is a tool, or mediator, for the communication actions performed. Therefore it is inherently social by nature and so can be regarded as a technically implemented social system (Goldkuhl and Lyytinen, 1982). Nonetheless, since it is technically implemented, an IS will physically consist of hardware and software that implements the required message processing.

The most important aspect of an IS, however, is its action potential, i.e. the repertoire of actions that it realizes. Actions can be performed by users in interaction with the system or by the system itself. In Chapter 4 three different types of actions performed in relation to information systems were identified: interactive actions, performed interactively through the system; automatic actions, performed by the system; and consequential actions, performed based on information from the system. The artefact’s interactive action potential is realized through its user interface (referred to as screen documents). Automatic actions can be realized through other sorts of documents, such as electronic data interchange (EDI) documents. An information system usually contains an action memory, commonly realized through database technology. This memory (which is part of the total organizational memory) can be used to remember what actions have been performed by and through the system. Therefore it serves as an important prerequisite for action and thus can be considered an important complement to the cognitive base (cf. Norman’s (1988) distinction between ‘knowledge in the head’ and ‘knowledge in the world’).

14.4.4 Information System Users as Business Actors

The actors participating in the communication actions performed in relation to an IS constitute its users. Actions ‘in relation to an IS’ can be performed through the IS, based on information from the IS, or by the IS. In Chapter 4 a distinction was made between three meta-roles of users. These are referred to as the communicator, the performer, and the interpreter. The communicator is the one that is responsible for the action and the resulting action-relationship established with interpreters. In doing business, it is quite possible that someone else performs actions on behalf of the communicator – thus acting as an agent. This is most evident in the case of automatic action where the artefact itself performs action. It might also be that one person performs action on behalf of another, however. For example, a salesperson who communicates an offer to a potential customer does this on behalf of the sales department, not as a private person.

14.4.5 The Business Context

All communication actions must be understood within the context in which they are uttered (Clark, 1996). Knowledge of the context helps in understanding a speech act.
Or put another way, a lack of knowledge of the context often means that a perfectly explicable speech act appears to be meaningless. This is so because the context implies that some references might be pre-supposed and thus need not be stated explicitly. An actor is required to understand the context to be able to successfully participate in communication. Stamper (1996) even states that ‘To give the pragmatic information an operational form we need to define the context in which signs have their effect.’

When doing business, the context of the communication is a business context. This means that, for example, some ethical standards as well as a basic understanding of the different actions involved in doing business can be taken for granted. Goldkuhl (1998) describes a phase model of generic actions performed by suppliers and customers in business processes, and several other such generic business models exist, for example, Action Workflow (Denning and Medina-Mora, 1995), SAMPO (Auramäki et al., 1988) and DEMO (Dietz, 1994; 2001). Such generic models exemplify what can be taken for granted in a business context. For example, both parties are supposed to know what constitutes an offer, what constitutes an order, et cetera, and what obligations are involved in communicating these different action modes to the other party. What is important from an actability perspective is that information systems successfully communicate the type of actions that are performed when using the system, and hence, the current phase of the process. Even though such a model serves as a good example for delineating a business context, actability is not restricted to commercial transactions alone. The term ‘business’ should rather be interpreted as any sort of organized collaborative behaviour aimed at some articulated goals. It is therefore highly relevant to talk about actability in businesses such as healthcare, public administration, and possibly other non-profit organizations.

14.4.6 Summarizing the Analytic Framework

The analytic framework consists of the four basic categories of the A³ model presented in Chapter 1: the actor, the action, the artefact, and the business context. The actors are the humans performing and interpreting action through and by means of the artefact. These are related to each other in two different ways. Firstly, there is a relationship between the actor, the action, and the artefact. That is, all three components must be understood as parts of a whole. The solid ternary relation in Figure 14-5 depicts this trinity.

![Figure 14-5: The four main categories of the analytic framework (Ågerfalk, 2001).](image)

Secondly, there are additional binary relationships between the actor and the action, the actor and the artefact, and the action and the artefact, indicating properties that are generic for actors in relation to artefacts regardless of what actions are per-
formed. The dashed binary relations in Figure 14-5 depict these less central relations, from an actability perspective. Additionally, all these relations must be understood within the particular business context for which the information system is designed. Note that the three meta-roles of users are distributed over the categories of actor and artefact, as described above.

The performance of actions by actors through artefacts can be further analysed with respect to the artefact. This is where the semiotic framework comes into play. Figure 14-6 shows how the actability concepts discussed above fit into the actability interpretation of the semiotic framework.

Figure 14-6: Main categories of the artefact part of the analytic framework.

Note that the category named validity in Figure 14-6 actually consists of the three sub-categories comprehensibility, truthfulness, and sincerity, while rightness is primarily related to the cognitive base (outside of the artefact), as discussed above.

14.5 Internet-Based Systems According to Actability

In this work, ‘Internet-based system’ is defined as a software-intensive artefact built by use of Internet technology. (‘Internet technology’ refers to the variety of technologies used by TCP/IP-based internets in general and the Internet specifically.) Thus, Internet-based system embraces everything from trivial static websites to full-blown information systems delivered via the Net.

The structure of this section follows the categorization presented in Figure 14-5 and Figure 14-6: it starts with the actors and follows on with the artefact. Additionally, some aspects related specifically to the development of Internet-based systems are discussed in Section 14.5.2.3. Action is considered in relation to the other two categories (the actors, and the artefact) throughout the text, and the business context is referred to where appropriate. It is beyond the scope of this chapter to go into details of the characteristics of Internet-based system. The aim of the chapter is rather to exemplify findings and to show how actability has been used to direct attention during data collection and abstraction (analysis).
14.5.1 The Actors

From an actability perspective, the obvious point of departure for understanding Internet-based systems is to direct attention towards the actors to whom it is supposed to deliver something valuable. In doing so, two primary categories of actors in relation to Internet-based systems were found: the host organization (i.e. the owner of the artefact), and its users (i.e. the actors using the artefact). Note that both of these can take on each and all of the three user meta-roles (see Chapter 4 and Section 14.4.4) at different times.

The users can be further categorized into three different generic target groups: internals, partners, or the general public. Internals are members of the host organization, partners are in a dyadic, usually contractually-based relationship with the host organization, and the public consists of actors to be identified outside of the host organization with no previous formal connection to it. At first glance, these three types of users seem to correspond to the three commonly used classes of Internet-based systems: the intranet, the extranet and the internet. However, in order to understand this classification appropriately, we must also consider the relation between the host organization and the users in terms of delivered action potential and required authentication. Either the system offers a controlled (restricted) environment, or it offers a completely open and insecure environment. Table 14-1 shows how these concepts relate to establish a more solid foundation for discussing different types of Internet-based systems.

Table 14-1: Generic target groups and methods for connecting to an Internet-based system (Ågerfalk et al., 2002a).

<table>
<thead>
<tr>
<th>Controlled environment</th>
<th>Internals</th>
<th>Partners</th>
<th>The Public</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intranet site</td>
<td></td>
<td>Extranet site</td>
<td>Internet site</td>
</tr>
<tr>
<td>Open environment</td>
<td>Internet site</td>
<td>Internet site</td>
<td>Internet site</td>
</tr>
</tbody>
</table>

Different target groups have a different understanding of the propositional content and action modes in Internet-based systems. These actors have different opportunities to use the system and different reasons for using it. Thus, the purpose of the artefact, the intended action potential, should be the guide for deciding its target groups. A comparison with traditional systems reveals a usage context as well. The main difference is the target groups. If transforming the concepts described above into traditional user categories, internals relate to ‘in-house’ development and the public relates to ‘off-the-shelf’ development. The partner category receives greater attention in the development of Internet-based systems because these artefacts have opened up opportunities for deeper co-operation with other businesses. The combination of user categories using the same artefact can be quite complex in Internet-based systems. A specific system can, for example, be partly an intranet-site and partly an Internet-site.

When applying this actor classification to the industrial partner, the important thing appeared to be not the classification as such, but the way it helped an understanding of two important factors influencing the possibilities of meeting user requirements. We may refer to these two factors as the level of knowledge about, and the possible power over the users that the host organization possesses in relation to the particular target group that has been identified (see Figure 14-7).

When it comes to public actors, developers can only estimate and make an educated guess about the character of the actors who will use the artefact. Under these
circumstances developers face a situation where their knowledge of, as well as their power over, the users is low. In the case of a system that addresses partners, the level of control can still be fairly low. Nonetheless it is possible to define an exact target group more easily, and therefore to gain greater knowledge about the users. A developer can therefore adapt more efficiently to the needs of specific users. Even when it comes to internals, the knowledge about users may be low, especially in a large multinational corporation such as our partner. The possibility of utilizing power over users is, however, more evident in this context and a possible lack of knowledge may not be so critical when it comes to designing an Internet-based system for internal use.

Figure 14-7: Power over and knowledge about users affect the development situation (Ågerfalk et al., 2002a).

As shown in Figure 14-7, if power is utilized by the host organization, the possibility of pursuing the development is at its highest. Either knowledge is attained through the power (the arrows in the figure), or the power is used to implement the system without respect for the users or their situation.

Development of traditional information systems faces much of the same challenge with regard to requirements engineering as does development of traditional systems. A difference is that Internet-based systems are easier to deploy and might therefore reach a more heterogeneous and remote target group. However, the threshold for reaching a competitor is lower for Internet-based systems than for traditional ones. Therefore, this situation is more sensitive.

14.5.2 The Artefact

14.5.2.1 Human Information Functions

An important property of information systems is their possibility to remember what has been said and done via their action memory. Actors both inside and outside of the host organization can affect the action memory. Actors outside the organization, partners, and the general public, have direct access to the action memory without going through internals. Thus, Internet-based systems allow for a greater impact on the host organization than traditional systems. Webshops are one example where almost all supplier
actions could be automatic actions. In this case, with the general public as the target group, the design of the action memory should suit an ‘unknown’ user.

An action performed by an actor consists of a propositional content embedded into an appropriate action mode. Both aspects are visualized through the system and can be made more or less easy to grasp. Users from different target groups differ in their knowledge of the business context. Therefore, their understanding of business actions supported by the system differs. Given the target group, it is still necessary to find a common denominator, in order to deliver intersubjective action potential. Otherwise, for example, actors using a webshop might not be aware at what stage they have actually committed a financial transaction. The obligations of such a commitment must be obvious to the user. Thus, the visual representation and the propositional content of the system becomes more crucial than it is in traditional systems, even though less may be known about the users.

Internet-based systems are easily deployed to a wide range of users. They may be distributed worldwide at a low cost. Then, differences in cultural background and belonging must be considered with respect to action mode and propositional content. Furthermore, there are differences in conceptual models (Norman, 1988; Orlikowski and Gash, 1994) regarding the business context and the consequences of business commitments. Therefore, each system could involve versions adapted for local business contexts. In these cases, actors need to be categorized further within each target group presented in Table 14-1. This elaboration must be based on the characteristics of each specific project (Karlsson et al., 2001).

One aspect of human information functions put forward as specifically important in relation to Internet-based systems development by our industrial partner is that of information security. From an actability perspective, information security relates strongly to the concept of validity (especially truthfulness, sincerity, and rightness) (cf. Ågerfalk et al., 2000b). In order to accept an action as truthful, the communicator and the interpreter must share a common belief regarding the state of the world. This seems to be a crucial aspect of Internet-based systems, since a heterogeneous target group with possibly varying social and cultural backgrounds are very likely, at least in a public context or, again, in a large multinational corporation. This need for intersubjectivity is of course important in order to understand the propositional content as well as the action modes communicated through the system. What about sincere intentions, then? This topic relates strongly to the very important concept of trust, from an Internet-based system perspective (e.g. Friedman et al., 2000). Trust is an important part of the cognitive base. Depending on the design of the artefact it is possible to emphasize this characteristic (Ågerfalk et al., 2000b). For example, a shift from an open to a controlled environment potentially yields a higher degree of trust. Trust can be related to information security. In turn, information security relates to the actability concept of validity (especially truthfulness, sincerity, and rightness), and thus both relate to the cognitive base and the pragmatics and semantics of actions. Of course, the importance of trust is reduced when users have a more formal connection to the host organization. If use of the artefact is purely internal to the host organization, trust is built through additional channels. Depending on the target group, trust is emphasized more in relation to Internet-based systems than in traditional systems.

14.5.2.2 The IT Platform

Software artefacts are realized through their IT platforms. Actability helps us to distinguish between the screen document and the message processing as two distinct high-
level categories. Screen documents are the GUI of the system, while the message-processing part is the software and hardware needed to distribute and execute interactive actions through the artefact as well as performing automatic actions. Since focus in a distributed system is on deployment and processing, it seems better to talk about deployment architecture instead of the more abstract ‘message processing’, constituting the total configuration of information technology needed to make Internet-based systems available to users.

The screen document should visualize the interactive action potential of the information system. Since the Internet-based system may be the only channel for business actions between actors, who might be novice users, it needs to be intuitive. Design of an ‘intuitive’ user interface is often based on one or more metaphors, relating to a concept already known to the users. A study of Internet-based systems reveals that the page-metaphor is the main metaphor for systems deployed (at least partly) as websites. Within the choice of ‘base-metaphor’, different business-specific metaphors may be found, such as the ‘shopping cart’ used in many webshops. The combination of metaphors should be based on the artefact’s target group. With a more heterogeneous actor group we can relate to the discussion about the human information functions in finding a common denominator in a metaphor. The use of different metaphors depends on possible techniques available in the deployment architecture. The design of the system is restricted by the host organization’s GUI standards. Since in many cases an Internet-based system is a publicly available artefact, it has to promote the host organization’s identity. The identity comes from the host organization’s perceived image in other business arenas. Via the design, this characteristic of Internet-based systems indirectly restricts the mood that screen documents communicate to users. The mood of the artefact is the interacting user’s first impression. Hence, it is important in an environment where alternatives might be ‘one click away’. This characteristic is most important with a public target group.

An important example of where Internet-based systems distinguish themselves is that they range from static Web pages (so-called brochureware) to full-blown information systems. By focusing on the delivered action potential, it has been useful to talk of an Internet-based system’s degree of interaction. The artefact can be classified according to two orthogonal dimensions (see Figure 14-8).

![Figure 14-8: Different degrees of interaction provided by an Internet-based system (Ågerfalk et al., 2002a).]
First, in the vertical dimension we find the terms ‘dynamic’ and ‘static’, which refer to whether the information presented is assembled in run-time or stored ‘as-is’ (for example, static HTML). A dynamic presentation could, for example, be a WAP-site depending on ASP-technology. Second, in the horizontal dimension we can classify the artefact as non-affectable or affectable. ‘Affect’ is used in this characteristic to indicate that use of the system can affect the action memory. Internet-based systems could have a purpose seldom found in traditional systems – information publishing. This case is found in the bottom-left corner in Figure 14-8. This type of system has the lowest possible degree of interaction. At the other extreme is an Internet-based system founded on run-time assembly of information that is affectable, such as a typical webshop.

As stated above, an Internet-based system’s deployment architecture constitutes the total configuration of information technology needed to make the system available to its users. Using the Internet as a platform for information systems implies communication between servers and clients using a network connection. The three principal components needed to create interaction with an Internet-based system are: a server containing the artefact (or the software assembling the artefact), a network connection, and client configuration which is capable of handling the delivered system (or parts of it). Therefore, we can divide the deployment architecture into three types of configurations: the server configuration, the network configuration, and the client configuration.

Each configuration consists of diverse sorts of hardware (at the physical level) and software (at the syntactic level). In the server configuration we find servers, databases and middleware/brokers. The client configuration often consists of PCs, browsers and software for communication with server parts. Network configurations are realized through network hardware and communication software. However, choices from the range of technologies cannot be made independently of each other. It all boils down to mixing these technologies with the demands from the required human information functions and the design of screen documents to create desired action potential. Compared with TSAs, we have less control over the client configuration, and have a more complex mix of technologies to balance.

The choices in each configuration affect the technical security and performance, and in that way affect an actor’s overall trust in the artefact (Ågerfalk et al., 2000b). Technical security is a brick in building trust at the cognitive base. Using a secure protocol for transferring data used by the system is one way to show that the host organization is trustworthy. Considering performance, each part in the deployment architecture has its own performance. Together they give the total performance of the artefact. Performance depends highly on where different parts of the system are executed. It is, for example, common to talk about thin clients, thick clients and distributed components (Conallen, 2000).

Each part of the system, screen documents, middleware, et cetera, has to be realized with one or more programming languages. Significant for Internet-based systems is the need to integrate different kinds of programming languages. The situation depends on possible needs to integrate with legacy systems, the server platform, the power of the clients, and the strategy for deployment. For example, a WAP-service cannot today rely on client-side execution.

The combination of hardware and software can be measured with respect to performance (at the empirical level). Each configuration (server, client and network) has its own performance. Together they give the total performance of the architecture. The performance limits the possibilities of supporting different types of architectural styles, such as thin or thick clients (cf. Conallen, 2000). An important aspect of deployment
architecture from an actability perspective is that the performance of the deployment architecture must match the artefact’s required action potential. For example, for some types of action the response time could be crucial and hence has a direct effect on the action potential.

14.5.2.3 Developing Internet-Based Systems

Three aspects of Internet-based systems that have been shown to be central for the development of such systems can be referred to as time to deliver, legacy integration, and newness of this type of artefact.

The time to deliver is related to ‘the need for speed’ generally characterizing the development of Internet-based systems. Internet-related development processes are often surrounded with the nimbus that Internet-based systems should be delivered even more rapidly than ‘traditional’ systems just because they are Internet-based systems (Overmyer, 2000). The requirement of aggressive release dates can be justified if the actions supported are business critical. With other and more non-competitor-related purposes, the time to deliver should not necessarily be stressed more than in the development of traditional systems.

Another property of Internet-based systems is that they often need to be integrated with legacy systems. An Internet-based system is often an interface-shell around a legacy system. This integration may call for advanced mapping activities between different artefacts. In a large information system context, this requires an interaction between many actors and interests during the development process. This may increase the complexity and risk in the development project by opening up existing systems to new user categories (partners and public).

The newness of the artefact can also introduce complexity and risk into the development. Development of Internet-based systems is to a large extent similar to traditional ISD. However, in the development of Internet-based systems, the construction of the Graphical User Interface (GUI) is often confused with the complete development process. Graphic design is an important aspect of Internet-based systems design, but it is only part of the entire development process. This confusion could be due to the fact that contracts in the projects being studied, have often been based on dummy GUI prototypes. The critical issue is to combine traditional ISD practices with Internet technology and artistic work, that is, to create a greater bond between artists and engineers. This reality can make execution as well as management of Internet-based system projects even more complex than for traditional development projects.

Other aspects of newness are linked to the techniques used to realize Internet-based systems. If these tools have low maturity (are insufficiently tested, lacking documentation, have uncertainties concerning support, et cetera), complexity and risk in the project might increase as well. Another facet of the problem is related to the development teams’ experience with these new tools, techniques and the Internet as a foundation for development as a whole. If the development team is lacking in competence, a seemingly non-complex project could be permeated with obstacles and uncertainty. Inexperienced teams, requiring training in these development techniques before and during the development of the projects, raise the demands for thorough preparation and support. These teams are also likely to need more time to complete projects than more experienced development teams. Inexperience can increase costs and delay completion of the project. This contradicts the need for Internet-based systems to be delivered quickly, but inexperience as a cost driver is not unique to Internet-based systems.
14.6 Concluding Actability Study I

This chapter has discussed and exemplified the operationalization of actability into a descriptive analytic framework. The analytic framework has been used to characterize Internet-based information systems, primarily as a means to ground actability empirically.

The framework has helped to identify differences in emphasis between Internet-based systems and traditional systems, but also to see that the characteristics themselves are not unique. Traditional systems have a usage context just as do Internet-based systems, but with different target groups. Internals can be thought of as traditional ‘in-house’ development users, and the general public as ‘off-the-shelf’ development users. In order to deliver ‘correct’ action potential it is important to understand who is going to use the artefact. Thus, just as in traditional projects, understanding the users’ requirements is important, but it differs in prerequisite and in focus. The target group could, for example, make it difficult to manage the combination of low knowledge of and power over the users, the need to design for trust, the right mood and intuitive GUI, et cetera. These characteristics must be managed through an accurate deployment architecture, which can vary greatly in configuration and performance between target groups. This is different from traditional systems where you can prescribe a certain configuration in order to guarantee a certain performance. The degree of interaction affects the need for requirements engineering. A static website presenting information could mean less focus on requirement work than usual. However, a dynamic and affectable Internet-based system is as complex an artefact as a traditional one. In such cases, requirements work cannot be reduced. Since Internet-based systems often involve legacy system integration, requirements engineering could involve a great deal of reverse engineering, but so can development of traditional systems; they also inherit restrictions, as well as potential benefits, from legacy systems. Aggressive release dates have been proposed as one of the more significant characteristics of Internet-based systems development. These can be justified in projects involving business critical actions, but such projects exist in traditional ISD as well. In the case of an Internet-based system, however, the artefact could itself constitute the whole business. Even though Internet-based systems are ‘hot’ and built with new technologies, we must remember that so were traditional systems but a few years ago.

The work shows that actability can serve as an effective tool in directing attention to communicatively oriented aspects of information systems. One important contribution from actability is an understanding of the different types of actors involved in the Internet-based systems context. This understanding has served as the baseline for the understanding of Internet-based system characteristics. The concept of ‘degree of interaction’ is, for example, one important concept for the characterization of Internet-based systems that is derived from the understanding of the actors and their required action potential, as is the relation between ‘power over’ and ‘knowledge about’ users.

In addition to actability, the semiotic framework has been used as an important inspiration. That way, an analytic framework that balances the human aspects of information systems with the more technological aspects has been arrived at. It seems the real strength of the actability concept lies within the conceptualization of, and in the relation between, actors, artefacts and actions. By utilizing the stricter hierarchical partitioning imposed by the semiotic framework, even more was achieved with actability. It also helped in focusing on different aspects more independent from each other when it comes to the artefact. Applying actability in accordance with the semiotic framework
was not as easy as we first thought. There are important conceptual and terminological differences that needed to be explicated and overcome before the analysis could be performed. Another problem encountered was the sometimes ‘not so obvious’ mapping of empirical categories to distinct semiotic levels. It proved to be the case that some categories, such as ‘technical security’ could not be related to one single level without separating it into possible sub-components, which were outside of the scope and purpose of the work. These problems also led to a continuous redefinition of the analytic framework. For example, the allocation of the concept of validity to distinct semiotic levels was not very obvious and led to further elaboration of the relations between trust, information security, and validity.

Altogether, the analytic framework has proved useful in finding and relating different important categories that characterize Internet-based systems, and the usefulness of actability has thus been further empirically validated in this study.
Chapter 15

Study II: Multi-Channel Interplay in Practice


15.1 Introduction to Study II

Lessons learned from the short history of e-commerce induce a focus shift from purely digital actors towards hybrid approaches (Steinfield et al., 2001). The hybrid approach implies a need for understanding how to utilize the complements between different marketing channels. A marketing channel is often considered to be a structure providing the means for all activities necessary to the process of furnishing an end-customer with some product. The channel concerns all actors involved, from manufacturer to end-customer. In this chapter, the focus is on the latter phases of such a process. The local electronic marketplace (LEMP) structure is Internet-based and concerns consumer-oriented business activities that are performed by actors using a digital (IT-based) as well as a physical channel. The LEMP context implies creating e-business means for local retailers to complement their physical stores.

Since actability is a concept for the understanding of IT-based information systems that emphasizes human actors and their performance of social action through and by means of these systems, it lends itself well to the exploration of the LEMP. By taking actability as a point-of-departure, the chapter shows how a deeper understanding of the interplay between multiple channels in LEMP practices can be reached – to wit: a deeper understanding of the LEMP multi-channel interplay in practice. Specifically, the aim of the chapter is to show how and why the synergies of a multi-channel structure are key success factors for the LEMP. This aim includes elaboration on how the actability of the IT-system, which constitutes the backbone of the LEMP, comprises a basic condition for the understanding and optimal utilization of this structure. A secondary purpose is to show explicitly how actability can be used to gain a better understanding of phenomena, such as the LEMP, that are related to information systems – a further enquiry into the empirical grounding of actability.

The chapter proceeds as follows. Firstly, the research method adopted is described; this includes an outline of the utilized analytic framework and a case description. Secondly, the hybrid channel structure of the LEMP is characterized as a practice for synergies between multiple channels. Thirdly, the interplay between channels that achieves such synergies is discussed from the perspective of information systems actability. Finally, the chapter concludes with a discussion about the benefits of viewing the LEMP from the perspective adopted in this dissertation.
15.2 Actability Study II Research Method

The research method adopted in this study can be characterized as an ‘explorative qualitative case study approach’. A case study is an empirical enquiry that investigates a contemporary phenomenon within its real-life context, which is especially useful when the boundaries between phenomenon and context are not clearly evident (Yin, 1994). This has certainly been the case in this work, since the focus has been on IT-systems within a LEMP-context, which involves a mixture of traditional business with e-commerce activities. The context also implies an inter-organizational setting that precludes a delineation based on company borders. This intricate issue of delimitation and an absence of a large body of related work are determinants for this study’s exploratory nature (cf. Benbasat et al., 1987).

To investigate the LEMP, two enquiries have been conducted. The first involves a qualitative, exploratory study of 30 Swedish websites with a local focus. The latter consists of two LEMP case studies of websites with a geographically local focus. This study included interviews with actors such as systems designers and retailers. The major empirical sources are, accordingly, online site content and 25 interviews. Using a qualitative approach was a natural choice since the aim was to understand how the LEMP is structured as well as how different ‘LEMP-actors’ think about and understand this way of doing business. The study emphasized website content, aiming particularly at retailers inhabiting the marketplace. The main empirical source is interviews with representatives from the companies hosting the marketplaces as well as from a criteria-based selection of retailers. Selecting respondents based on certain criteria rather than, for example, randomly, makes it possible to ensure the relevance of the data collected and the final results. The criteria include, for example, a requirement that retailers should utilize both a digital and a physical presence; that the digital channel be used for sales interaction (not restricted to presentation) and that the retailers be comparable with respect to size and marketed product types.

A major influence during the early phases of this casework has been the notion of practice (Goldkuhl and Röstlinger, 1999) as it has been used as an illustrative theory for collecting the qualitative data. That is, one presupposition of the study is that the LEMP can be viewed as a practice (defined in Section 15.2.1), and that data can be collected on this basis. This standpoint implies viewing the electronic marketplace as the result of actor co-operation effected by a diversity of intentions (Petersson, 2001a).

For the analysis of the data collected, an approach utilizing an analytic framework based on the concept of information systems actability and the notion of practice is adopted (described in detail below). By using an explicit framework, the analysis becomes focused on certain aspects that are believed to be important (Patton, 1990). This is in contrast to a strictly inductive approach, such as the Grounded Theory of Glaser and Strauss (1967), and favours a reflective approach in which theory is allowed to evolve as it is being used actively in the research process (see Chapter 3). Specifically, the analytic framework has been used deductively to find relevant issues in the empirical data, and to explain identified phenomena inductively. Deductively, the data has served mainly to vindicate the usefulness of the actability concept inherent in the framework, and the framework has been used to direct attention to, and to provide explanations for, the occurrence of the phenomena. Inductively, the data has served mainly as a source for identifying phenomena and the framework as a means to understand and classify these.
One of the strengths of the framework, and of its theoretical base, is that it is general enough to be used to explore IS-related phenomena without having to pre-specify in detail the borderlines between phenomena and context, which was a pre-requisite in this case. The generality also means that the framework is scalable in that it can be used to handle the various levels of abstraction that are needed to understand a complex socio-technical phenomenon such as the LEMP.

15.2.1 An Elaborated Analytic Framework Based on Actability

The first part of the analytic framework adopted is explicitly derived from the concept of actability. This part of the framework corresponds to the four main categories of the analytic framework used in Chapter 14: the actor, the action, the artefact, and the business context (see Figure 14-5). In Chapter 14, this part of the framework was used as a means to understand and discuss Internet-based systems in general. In this work more focus is put on the work practice (i.e. the business context) in which software artefacts are used to create a LEMP. For this purpose, the framework has been extended with a general notion of work practice (Goldkuhl and Röstlinger, 1999). Here a ‘practice’ is considered to be a ‘doing’ that is not necessarily limited by company borders. A practice is a performance of actions and is defined as follows: ‘A practice means that some actor(s) – based on assignment from some actor(s) – makes something in favour of some actor(s), and something against some actor(s), and this action is based on values, rules, knowledge and competence, which are established and continuously changed’ (Goldkuhl and Röstlinger, 1999). Figure 15-1 shows a conceptual map of the analytic framework.

![Conceptual overview of the analytic framework](image)

**Figure 15-1:** Conceptual overview of the analytic framework (the arrows indicate intended direction of reading). (Ågerfalk and Petersson, 2002)

The core of a practice can be isolated as the part of the ‘doing’ that involves refining some input into a desired kind of output. This process of transformation is central, but should also, according to the notion of practice, be comprehended as highly de-
pendent on various assignments and social norms. A dominant view on business processes suggests focusing external assignments to create ‘customer value’. Internal assignments should, however, also be acknowledged as affecting the result for all practices (commercial as well as non-commercial). So-called ‘role assignments’ concern the conditions under which different producers act. Actors benefiting from the final product are called clients. (Goldkuhl and Röstlinger, 1999) The notion of practice goes well with the concept of information systems actability as they are both founded in similar theoretical notions and thus share many core concepts (Goldkuhl and Röstlinger, 2002).

15.2.2 Description of Marketplaces Studied

As indicated above, a case study of two LEMPs with a geographically local focus has been conducted. The objects of study were the Swedish websites Lokaltidningen62 (LT) and Skaraborg-Online63 (SOL), which can both be categorized as local electronic marketplaces (Petersson, 2001a).

Lokaltidningen is a Web-based newspaper that has expanded into an electronic marketplace that targets a rural district in Sweden. The site first of all provides news produced by an editorial staff located at a small advertising/Web agency. The site also publishes other kinds of local content (for example, municipal information, an online music archive and organizational activities) produced by different categories of site users. An essential feature is LT’s discussion forum, which concerns everyday life and focuses on local themes, such as current events and politics. LT also includes a section that provides an electronic mall of webshops. The mall is inhabited by retailers and based on a shared IT-system developed and maintained by the Web agency. The typical LT retailer is a small company also running a physical store located in the area. These retailers offer products in various categories ranging from home electronics to groceries.

The other marketplace, SOL, is similar in scope and basic ideas to LT. SOL is, however, more of a true joint venture, based on co-operating parties. The site targets another geographical district in central Sweden and is co-ordinated by a small Internet firm. The most prominent partner is Torget.se, a nation-wide electronic marketplace with a consumer focus. This venture directs a locally targeted section of the national marketplace that contains an electronic mall of retailers in the same manner as LT. SOL presents local news feed from another partner and, like LT, the site hosts a discussion forum. The site also hosts a ‘search engine’ committed to local Internet content.

15.3 The Multi-Channel Interplay of Local Commerce

When addressing electronic marketplaces in a general sense, there are various definitions in the literature. In Bakos (1991), an electronic marketplace is described as enabled by an interorganizational IS that ‘allows the participating buyers and sellers to exchange information about market prices and product offerings; thus it represents an investment in multilateral information sharing’ (Bakos, 1991). An IT-based information system constitutes the backbone of any electronic marketplace. It is also important to note that a particular electronic marketplace does not necessarily offer digital channels for all phases of business activities. A combination with physical channels is, on

---

62 http://www.lokaltidningen.net
63 http://www.skaraborgonline.com
the contrary, often a prerequisite for the mere existence of an electronic marketplace (Bakos, 1998).

15.3.1 Electronic Marketplaces for Local Commerce

Presenting the concept of the electronic marketplace often involves discussing the needs fulfilled by the marketplace in terms of functionality (Bakos, 1998; Turban et al., 2000; Zimmermann, 1997). When it comes to consumer-oriented marketplaces with a geographical focus, it is possible to distinguish the twofold functionality of ‘market activity’ and ‘communication’ (Zimmermann, 1997). This dichotomy seems to be crucial for the understanding of this study’s empirical observations, but the terminology is somewhat misleading. From a social action perspective, any ‘market activity’ concerns communication between social actors, such as buyers making requests and sellers making offers. Therefore, the functionality can be expressed as a means for ‘commercially oriented’ versus ‘non-commercially oriented’ communication. Marketplace functionality concerns the facilitation of, for example, supply, demand and pricing (i.e. commercially directed functions), but also of actor communication that is not aimed at doing business (i.e. non-commercially directed functions).

A LEMP is enabled by an Internet-based information system and the targeted user group shares an interest in a specific locale (Petersson, 2001b). This type of electronic marketplace is a publicly available environment including business activities predominantly concerning multi-channel actors practising consumer-oriented commerce. Of course, it is not publicly available in the sense that anyone can do anything. Parts of the marketplace require authentication to enable access, and in this respect, at least, there exists a previous commitment to certain rules governing the LEMP.

15.3.1.1 Commercially Oriented Communication

Commercially oriented LEMP activities may concern business communication in ‘business-to-consumer’ as well as ‘consumer-to-consumer’ relationships. The latter may be realized by a subsystem handling classified advertisements. The business-to-consumer-oriented functionality is often organized according to a mall model (Timmers, 1998; Turban et al., 2000). Retailers exploiting digital channels as an extension of their physical appearance can be categorized as storefront holders (Saarinen and Tuunainen, 1998) adopting a hybrid approach to electronic commerce (Steinfield et al., 2001). A central idea is the use of both digital and physical presences with which to interact and meet customer needs.

Several arguments in favour of hybrid solutions instead of exclusively physical or digital presence have been put forward (Steinfield and Whitten, 1999). The main standpoint stresses possible synergies inherent in the utilization of dual channels. One argument relates to embeddedness and trust. Embeddedness has been discussed as the influence of personal relationships on business interaction (Granovetter, 1985). The argument is that economic rationality is embedded in the social relations of actors, which influences possible trust. The topic of trust is isolated as a critical aspect for the success of electronic commerce relationships (Timmers, 1999). Further arguments relate to increased possibilities in meeting diverse consumer needs and behaviour; exploiting the natural complementarity between virtual and physical capabilities to enhance value for buyers; and utilizing intrinsic knowledge in the local community.

A hybrid approach might be realized by different strategies that imply a variation in possible channel complementarities. Retailers can be seen as adopters of two differ-
ent strategies: the ‘mirror strategy’ and the ‘synergy strategy’ (Timmers, 1999). Adopting a mirror strategy here involves resembling the physical operation of the store. The digital channel then presents a mirror image, exhibiting the same products (or a selection of the same set) and the same bargains with no exploitation of possible complementarities. LEMP retailers adopting the synergy strategy handle the two channels in a more intertwined way. This might mean, for example, using the Web for mediating pre-purchase information and after-sales services, whereas the physical store is used for delivery (Petersson, 2001a).

15.3.1.2 Non-commercially Oriented Communication

There are two main LEMP features that fulfil the non-commercial function: community of local interest and informational services. The discussion forums constitute an online community (Preece, 2000) in which the shared purpose for participation is an interest in the particular physical community. The marketplace host often manages the community feature (see below) by raising different issues and themes for discussions to which visitors can contribute. Informatory services fulfil the non-commercial functionality by presenting information not produced by visitors but of interest to the local community. Examples include information on upcoming events, local news and weather forecasts.

15.3.2 Practising Marketplace

It is possible to describe a LEMP by outlining some typical actor categories (Petersson, 2001a). A host (host organization) co-ordinates the LEMP activities, managing the shared marketplace structure. This includes the host as system owner providing access to, developing and maintaining the IS. The visitor category consists of site users, the established and prospective customers of the inhabiting retailers. Since the motive for marketplace use is a local interest, a visitor may well be seeking information that is not related to commercial activities. The habitant is the ‘tenant’ of the marketplace, either as non-profit or commercial actor. The non-profit habitant might be a sports club or another local organization with a need for disseminating information. A retailer also running a physical store within the targeted area is the most typical commercial habitant. From the host’s point of view, this kind of habitant is also the main source of revenue. A content supplier is an external actor who, in concert with hosts, habitants and visitors, provides site-content; for example, services delivering weather forecasts or news. The rest of the actors steer the contributions from content suppliers by assignment.

As the LEMP serves other than commercial purposes, it can hardly be delimited by organizational borders. Adopting the notion of practice makes it possible to contextualize the actor categories and their relationships. The clients of the LEMP practice are actors benefiting from the use of its products (the result). Clients can be found among visitors, habitants and content suppliers. The LEMP as a product can be understood as a medium in which clients can perform desired actions – literally forming an action space. Using the marketplace IS should then be understood as utilizing the system’s action potential, which spans this space.

Visitor purposes are to become informed, to communicate with fellow visitors and to do business. Commercial marketplace habitants need the marketplace for marketing and sales, whereas non-profit tenants strive for dissemination of information. The third type of marketplace client consists of content suppliers also concerned with marketing.
Study II: Multi-channel interplay in practice

The **production** of this practice is the process of creating the LEMP’s action space – a process that should be understood as a co-production by marketplace hosts, habitants and visitors. The fact that the host serves as system owner accountable for IS functionality does not mean that actions performed by others have no effect on outcomes. The retailers and non-profit habitants design and operate their individual parts of the marketplace. Aspects of visitor use can also be understood as vital activities of production; visitors contribute to and shape, for example, discussion forums and classified ads.

15.4 Actable Systems for Interplay

In order to obtain the desired synergies between different channels, it is vital that all actors involved be comfortable with each of them and be able to utilize their integration. It is important to make both customers (as visitors) and retailers (as habitants) understand that the business context changes as new channels are introduced, which should be understood as an opportunity rather than an obstacle. The enabling IS is central for the utilization and optimization of the different channels within the LEMP. To understand the role of the IS, we must consider the action space that constitutes the LEMP. Utilizing the IS to achieve the desired synergies between the different channels within the LEMP is therefore to understand the social actions performed through interplaying channels.

15.4.1 Action through Multiple Channels

Business actions performed via the physical channel can be based on the digital channel (as consequential action, see above). It is therefore important to align the two channels so that potential synergies might emerge. A good example of this is a SOL retailer in home electronics that uses the physical store primarily to provide the look-and-feel of products, while directing customers to its website for hands-on installation instructions and in-depth product information. The latter provides opportunities for using existing resources by linking to, for example, manufacturers and independent evaluators. The LT case presented habitants dissatisfied with the absence of large-scale sales through the digital channel. These problems could be related to a lack of understanding of the potential for inciting customer actions through information from the IS. These problems might potentially be solved by providing the means for tracing physical in-store actions to website promotions.

The marketplace structure should be utilized to achieve synergies not only between the different channels, but also by intertwining the commercially directed and the non-commercially oriented functions. Associating tailored commercial information to appropriate discussion forums may be a way to achieve this. Another example is commercially directed adjustment of the online community, such as introducing topics more closely related to retailer offerings (Hagel and Armstrong, 1997). This kind of adjustment was discussed by the LT host but never implemented. The delicate task of controlling the online community is a matter of understanding and considering the social norms of the LEMP business context.

15.4.1.1 Validity of Communication

In order for communication actions performed at the LEMP to be successful, it is vital that social facts established within the multi-channel structure are communicated via the IS. This includes correlations between information communicated via the different
channels. For example, offers in a physical store may correspond to on-line offers, or commitments made in the physical channel may correspond to those made (and made visible) via the digital channel.

The study came across webshops maintaining out-of-date offers. The habitants behind these offers were sometimes aware of this, but did nothing about it, passing the responsibility to the marketplace host. The validity claims (see above), which imply that information presented via the IS should usually correspond to the actual capacity, range of merchandise, special offers, and so on, were thus not met in this case. The host, on the other hand, referenced the initial agreements stating the retailer’s liability.

15.4.1.2 Interactive, Consequential and Automatic Action

Interactive action refers to action performed through the digital channel. Web-based retailing has shown advantages over to its physical analogue from a transaction-cost perspective (Steinfield and Whitten, 1999), and one way of utilizing the digital channel can be to encourage customers to perform actions interactively rather than via the physical channel. Multi-channel actors can exploit this as well, and there is potential in directing selective offers to the digital channel. For example, LT promotes this direction from physical to digital by offering prices on the Web that are different to those in a physical store, and clearly exposing both prices.

Aligning the digital with the physical presence can allow a webshop to communicate business offers intended for customers to negotiate and eventually accept via the physical. The Web can be a means to get people into the physical store where salespersons better can provide for a customer’s individual needs. This situation can also generate additional sales. An activity exemplifying this purpose is LT’s visitor competitions, where the prizes are gift cards for use in the physical stores.

The IS’s capability of performing automatic actions can be highly utilized within the multi-channel structure of the LEMP. Information about actions (both digital and physical) stored in the action memory can, for example, be used to perform automatic business offers. This is particularly the case in business performed customer-to-customer through classified ads. One example of this is the LT facility whereby a search among private advertisements also generates results from associated webshops.

Utilizing automatic actions would also have been a way to minimize the exposure of out of date offers (as described above). Hosts could, for example, provide automatic IS routines to remove specified information at given points in time, or force retailers to acknowledge the validity of published information at regular intervals.

15.4.2 Utilizing the System

15.4.2.1 Action Potential

As described above, the functionality of the IS realizes an action potential that spans an action space of the LEMP. For the commercially directed functionality, this encompasses, among other things, a web mall’s internal as well as external facilities. It is important to see that creating these facilities, by designing the IS, is to create conditions for action. Internally, the structure of LT permits a habitant choice in using a standard solution or a tailor-made design. Of course, this means leveraging action space to different extents and accounts for a shift in action potential responsibility. Designing the external LT mall facilities, such as integrating the search results from the classified ads section with retailer offerings, is another example of manipulating the conditions...
for acting. This IS design as creation of marketplace action space should be acknowledged by the hosts and the marketplace IS designers.

15.4.2.2 Action Memory

The action memory of the LEMP IS can (and should) be used to store information about actions performed through the IS, that is, via the digital channel, as described above. This is important in order to maintain, for example, customer profiles and to enable direct marketing activities and personalized information supply, as frequently used at LT. To take advantage of the multi-channel structure of the LEMP, the action memory should not be restricted to this ‘traditional’ purpose. Rather, the action memory provides an opportunity to maintain information about activities performed, in whole or in part, via the physical channel as well. In this way, the digital channel can be optimized according to activities primarily performed via the physical channel, and vice versa. This opportunity, however, has not been utilized in the two LEMPs in this study.

15.4.3 Multi-Channel Actors

Within the concept of actability, there is a key distinction between performers, who perform actions, and communicators – those on whose behalf actions are performed. Actors within the multi-channel structure should therefore be aware that they sometimes are only performing, or mediating, someone else’s intentions and that sometimes other persons or IT-systems are performing actions on their behalf. This distinction is a key to understanding that offers made at a website are performed by the IS but communicated by the retailer running the particular webshop. This notion is clearly applicable in the case of outdated offers, as described above, in which the host should be understood to be responsible for the system’s action potential, but not for the actual actions performed. There should however be a mutual interest in maintaining the overall validity. With this understanding it is also clear that a salesperson’s actions must not only match actions performed by other salespersons, but actions performed by the different systems involved in the multi-channel structure as well.

15.5 Concluding Actability Study II

This chapter has described the local electronic marketplace (LEMP) as a practice enabling synergies between multiple channels for business interaction. The focus has been on physically established retailers adopting a hybrid approach for doing business with their customers. The hybrid approach has been argued as a favourable business strategy in that it utilizes the strengths of a physical presence integrated with a digital presence. This is of particular interest to a local setting where the proximity of a physical establishment can be a base for e-business trust (Ågerfalk et al., 2000b).

The analysis has been based on the concept of information systems actability and the notion of practice. From this theoretical base, the LEMP setting has been described as an action space realized by an enabling IS. IS functionality has been discussed in terms of action potential (as a repertoire of possible actions) that spans the action space constituting an arena for interaction, commercial as well as non-commercial.

The notion of practice has been used as a point of departure for outlining a phenomenon not easily delimited by organizational borders. Since the LEMP relies fundamentally on the IS upon which it is based, the concept of actability has been used to identify important factors for the success of a LEMP.
A key issue for multi-channel settings appears to lie beyond the digital transaction-cost benefits frequently discussed in the literature (Steinfield and Whitten, 1999). The issue concerns the view of business actions as embedded in a social context as a basis for trust (Granovetter, 1985). It seems that the approach taken makes it possible to identify these insights, and potentially makes it possible to implement them in the enabling information systems.

The results show that there is more to be gained in terms of synergy between the digital and physical channels related to the LEMP if these are viewed as tools for performing social action.

It is not satisfactory to restrict the use of the IS to utilizing the digital channel in order to improve it by, for example, collecting customer on-line behaviour. An ‘actable’ system should rather keep track of the history of actions taken in both digital and physical channels. This would give a better basis for optimizing the desired synergies. The impact of IS use becomes untraceable if one neglects actions based on information from the system, even those actions performed through the physical channel. Acknowledging this aspect of an IS’s potential contributes to the understanding of the LEMP’s business context as a whole. It seems commercially directed communication in a LEMP setting pushes claims of validity to extremes. As much as a multi-channel structure offers the potential for leveraging advantages, it can also increase damage. A bad appearance through one of the channels, of course, mars the overall image.

The benefit of using social action theories (in the form of actability and the notion of practice) as a base for understanding this multi-channel interplay is that it helps to identify the LEMP as a social practice spanning company borders, and provides guidance as to how to design information systems that truly support such a practice.
Part VI

Conclusion
Chapter 16

Summing Up and Planning for the Future

This chapter returns to the issues addressed in Chapter 1 and elaborates how these are treated within the concept of actability. The chapter thus summarizes what has been discussed so far in the dissertation and gives the author’s view of how the different parts are related, and on possible future actability research. Before going into details, the contribution of the dissertation is discussed from the perspective of perspective. This is important since the concept of actability should be understood as a contribution to how one may choose to view information systems – as an information systems perspective that emphasizes information technology as a tool for business action and communication.

16.1 Actability as a Perspective on Information Systems

As discussed in Section 2.2, a perspective is inherently applied when working according to a theory. Actability has in this dissertation been described as summarizing a theory of information systems as information action systems. An important contribution of this theory is that it promotes a perspective that emphasizes the social action character of information systems. A reasonable question to ask is whether the actability theory should be regarded as all-encompassing – as the theory with a capital T? Certainly no such theory exists, at least not within the information systems field. As pointed out by Winograd (1988, p. 4):

‘Within the community concerned with the design of computer systems, there is a growing recognition of the importance of the designer’s perspective – the concerns and interpretations that shape the design, whether they are articulated explicitly or are just part of the unexamined background of the work. A perspective does not determine answers to design questions but guides design by generating the questions to be considered… no one perspective covers all of the relevant concerns… although the full range of perspectives must eventually be considered, the outcome will differ depending on where we start.’

The perspective promoted by actability is but one of many possible. Actability puts on the agenda questions such as: who does what to whom, and what is required in order for that to be possible in ways that are satisfactory for the individuals and the organizations involved? Actability focuses on qualities related to the business use of information technology. Nonetheless, it has consequences also for internal ‘system oriented’ qualities in terms of, for example, design of databases as action memories and user interfaces as arenas for interactive language games. Following the above quotation of Winograd (1988), actability must in practice be merged with, or at least used in combination with, other perspectives focusing on other important aspects, such as software reliability, education, and software maintenance. The important thing is that, as Winograd (1988, p. 4) puts it: ‘the outcome will differ.’

Now, the question is: if the principles and techniques presented in this dissertation are followed, will the outcome differ? If it differs, then how will it differ? And what will it differ from? Inherent in the notion of perspective is the fact that emphasizing
certain aspects means de-emphasizing or even ignoring others. As a consequence, (1) you may not be aware of a potential difference should you not apply the perspective since you would not look for the things stressed by the unapplied perspective, and (2) you may very well achieve the same outcome regardless of perspective applied.

The first point is important since it implies that formulating and externalizing perspectives are vital for putting specific properties of phenomena – in this case the phenomenon of information systems – on the agenda, and thus making it possible to discuss different alternatives. The second point is particularly important since it relates to the relevance of this type of research altogether. If point 1 is sorted out and we agree that a particular perspective emphasizes important aspects, could we achieve the same outcome without applying it explicitly? If so, externalizing a perspective and operationalizing it into method support and descriptive frameworks would be highly irrelevant. Not surprisingly, since this dissertation has been written, this author believes that it is still highly relevant. The thing is that there is a tremendous difference between succeeding in something, and knowing why the success was achieved. Of course, one can live happily with a certain degree of ignorance and manage to go on and construct ‘licable’ information systems while totally unaware of this work, or information systems actability altogether for that matter. The thing is that by externalizing and operationalizing a perspective, it becomes possible to apply it consciously. Qualities promoted by a perspective – in this case that of actability – may very well be achieved by chance. On the other hand, by explicit recognition the same qualities may be achieved by conscious design. Arguably, actability by design is far more important than actability by chance. Potentially, designed outcomes, as compared to unintended ones, eventually lead to better practice and a way to learn more about the perspective one applies, which would then potentially lead to superior quality outcomes in the future. This way of thinking about consciousness in design practice relates to work on ‘capability maturity’ (e.g. Herbsleb et al., 1997), which stresses that a managed process in which one knows what one is doing, and is capable of improving it, is what to strive for.

16.2 Revisiting the Point of Departure

In Chapter 1, the work presented in this dissertation was framed by two partly overlapping paradigmatic foundations, referred to as contextualization and pragmatization of information systems. Contextualization of information systems refers to the recent move from an isolated systems approach towards a contextual understanding of information systems as parts of a larger physical and social context. In Chapter 1 this issue was discussed in terms of a broad concept of usability for understanding usage quality, a potential gap between business requirements and system requirements, a need for integrating cognitive and organizational approaches, and a clarification of the concept of an information systems user. The required pragmatization of information systems was derived from three different IS fallacies: the usage fallacy, the instrumental fallacy, and the descriptive fallacy.

In the remainder of this section, these paradigmatic foundations are readdressed and through an examination of how the introduction of the concept of IS actability may promote a conscious and reflective contextual and pragmatic understanding of information systems. Let us start by turning our interest towards the concept of usage quality to see how it may be understood from the perspective of IS actability.
16.2.1 Information Systems Usage Quality Revisited

Actability embodies a usage-centred view of information systems. This differs from a user-centred view since actability is not particularly focused on users but on users’ use of a system, within a social business context. Chapter 1 presented this usage-centred view as related to usage quality. Usage quality being the quality achieved when an IS satisfies stated and implied needs during use. As discussed by Bevan (1995a), the use-context ultimately determines the measure of a particular products quality in use or, synonymously, usability (in a broad sense). From the perspective of actability, usage quality is not a static property of a system but is contingent upon the systems’ use within its particular business context. The business context thus frames the social and organizational environment of an IS. The business context consists not only of individual task goals but also of actors, situated in time and space, performing communication actions and material actions, which must be related to an inter-subjective understanding of the business context. This inter-subjectivity is required since actability assumes that IS use cannot usually be understood simply as a user interacting with a system. Actability assumes that actors are doing intentional things that are oriented to the behaviour of other actors. From an actability perspective, an IS can be used as an interactive tool. This is only one limited use of information systems, though. Information systems are also used as automatic machines, performing business actions oriented to the behaviour of actors. Such automatic actions are always ultimately derived from rules predefined by humans. Among other things, such rules describe on whose behalf an automatic action is performed. That is, who is the communicator of a particular act performed by the IS? Taken together, this means that it is an oversimplification to speak only of a user’s interaction with a system. Actability suggests three meta-roles of users: the communicator, the performer, and the interpreter. The communicator is the one responsible for an action and corresponding action relationships created with interpreters. Interpreters are the ones interpreting actions and resulting messages handled by the IS. Sometimes an interacting user performs actions, thus acting as performer of interactive action. Sometimes the IS itself performs the action, thus acting as a performer of automatic action. Hence, the performer is the actor or the system performing action on behalf of the communicator. Sometimes the communicator is also an interactive performer. On the other hand, an automatic performer is always an agent for some human actor acting as communicator. Interpreters, in turn, may perform actions based on previous actions that have resulted in information in the system, which is interpreted by the interpreter.

These factors (actors performing intentional business actions directed towards other actors, by use of information systems interactively or as automatons, and possibly by delegation from someone else) ultimately determine the effectiveness, efficiency and satisfaction achieved in the use-situation.

Since we are dealing with a social action context, these criteria should not be interpreted only using an instrumental orientation. For an action to be effective it should establish desired action relationships and that establishment should be performed in accordance with accepted social norms in the business, in order to create a mutual understanding of the established social fact and the way it was established. For an action to be satisfactory the performance should be possible without discomfort and with

---

64 Note that the term ‘usage-centred’ is used in a similar but partly more restrictive way within what has been called Usage-Centred Engineering (Constantine and Lockwood, 2002). In our work we simply use the term to mean a focus on IS use rather than on IS users.
Information systems actability

positive attitudes towards the use of the system. Equally important, though, is the extent to which a user feels that what he or she is doing is trustworthy from the perspective of the intended interpreter, as well as to the degree he or she can trust information from the system on which to base action.

Let us now return to the paradigmatic foundations of this work and see how they relate to the actability understanding of usage quality.

16.2.2 Actability and Contextualization of Information Systems

During the last decade we have seen a shift in research on information systems and human-computer interaction, a shift from focusing IT-systems as isolated entities towards IT-systems as parts of a larger context (Kuutti, 1999, p. 360):

'It now seems to be generally accepted that designing the technical “core system” alone is insufficient, and that in order to design and implement a successful IS some kind of “context” has to be taken into account – a context that includes people and their relations'.

Among other things, this shift has had repercussions on the whole notion of a usable system, and hence on the understanding of the concept of usability. As a result, it is possible to talk about usability in (at least) two different meanings (Bevan, 1999). The first meaning represents a narrow product oriented view interpreting usability as ‘ease of use’. This view is reflected in, for example, the work of Nielsen (1993; 1994) as well as in the usability definition of ISO/IEC 9126 (Bevan, 2001, p. 537): ‘The capability of the software product to be understood, learned, used and attractive to the user, when used under specified conditions.’ The second meaning represents a broader view aiming at the ultimate goal of useful artefacts. This view is reflected in many recent accounts of usability and most notably in the usability definition of ISO 9241-11 (1998): ‘The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.’ Both of these meanings are important and useful when discussing the usage quality of an IS (Bevan, 1999; 2001). From the perspective of actability, usage quality is mainly related to usability in the broad sense. The actability of an IS is related to its functioning as a social business tool, not primarily to its user interface design. If the system does not permit, promote and facilitate desired actions, it is notactable. Nonetheless, appropriate design of the user interface is imperative in order to make the action potential of the system available to users. However, it is important to take into account the usage fallacy and the instrumental fallacy when interpreting the concept of usability (see Section 16.2.3).

In relation to the two views on usability for usage quality it is important to be aware of a potential gap between business properties and system properties – between business requirements and system requirements, in terms of requirements engineering. Even if usability is interpreted in the broad sense there may still be a gap between system properties and properties of the business processes the IS supports. One major risk is to focus on one single user’s performance of isolated tasks. From an actability perspective, users’ interactive tasks (or interactive use-situations) are often interrelated since performing a business action and interpreting the results may belong to different use-situations. Use-situations are the basic actability unit of analysis for understanding human-IS interactions, forming use-situations, and relating these to the business processes of which the use-situations are parts. This way, there is a theoretically justified
Summing up and planning for the future

This integration of modelling activities relates to the need for integrating individual (cognitive) and organizational approaches to business and IS design. On the one hand it is important to acknowledge different users’ perceptions of systems and business tasks. On the other hand it is important to frame the IS use within a firm understanding of the business and its processes. The actability answer to this need for integration lies in the understanding of e-interactions that form e-actions that form the action structures of business processes. Utilizing contextual descriptive techniques such as Actions Diagramming together with I-Tables have proven to be a workable approach towards the desired seamlessness – towards bridging the requirements gap. Utilizing early prototyping together with I-Tables and Action Diagrams is an important complement.

A further issue regarding contextualization is the actability clarification of the concept of IS user. From an actability perspective, users act as communicators and interpreters of information. Differentiating the communicator role from that of the performer has proved important in order to understand responsibilities within a business analytically. Who does what to whom on who’s orders? Accepting that information systems per se can perform action as agents for someone else puts another piece of the puzzle in place. This way IS use can be understood as delegation of action responsibilities, which acknowledges that business information systems are not always easily understood simply as ‘interactive systems’. Certainly, there are other stakeholders involved not explicitly dealt with by actability. The important thing, from an actability perspective, is a potential move from focusing system ownership towards focusing action responsibility.

16.2.3 Actability and Pragmatization of Information Systems

Taking up a broad approach to information systems and acknowledging the use context as important does not necessarily mean that information systems are viewed as pragmatic tools for business actions and communication. The ‘usage fallacy’ refers to the situation when IS use is in focus, at the expense of focusing the business actions that make the use relevant in the first place. That is, when unreflectively assuming that people want to use a tool rather than getting things done by using it. Even though actability promotes a usage centred perspective, its point of departure is the business actions performed when using information systems. There are other important results and prerequisites in a use-situation apart from individual task goals and individual and technical abilities and constraints. There are important goals in terms of intended social consequences, such as established social facts and commitments. There are other functions of social action than achieving easily graspable goals, such as conforming to established social norms. Taking pragmatics as starting point means to focus on pragmatic aspects such as actors, goals, norms, and actions as a means to understand IS use, rather than focusing on IS use as a means to improve IS design.

This also means that it is insufficient to value performance of social action from the perspective of an instrumental rationality alone, which would be an instance of the ‘instrumental fallacy’. Since business actions are performed within a business context of other actors with their own set of expectations, goals, norms, et cetera, business action can, to a great extent, be seen as communicative action. For communicative action to be successful, communicators raise certain claims of validity and presuppose
these to be accepted by interpreters. Such claims are related to the possibility of being understood, expressing sincere intentions, referring to an inter-subjective understanding of the business and conforming to established norms.

Finally, pragmatization of information systems means a rejection of the simplistic view that information systems only contain descriptions of reality to be used by users to get informed. This restrictive view of information systems is an instance of the ‘descriptive fallacy’ and may lead to systems that do not contain relevant information and that are hard to understand, use and maintain. Information systems are information action systems. Such systems should, from an actability perspective, permit, promote and facilitate business actions. This includes storing information about actions performed and other important action prerequisites. Therefore, viewing information systems as images of reality and not as important parts of reality, may lead to a lack of information about important business actions in the system, as well as a lack of possibilities to perform desired business actions through or based on information from the systems.

16.3 Does Actability Reinforce or Reinvent?

The basis for the concept of actability is the paradigmatic foundations discussed above. These include knowledge mainly related to usability, communication modelling (i.e. language/action and semiotic perspectives), requirements engineering and conceptual modelling. The concept of actability has evolved through a re-interpretation of these and other knowledge domains from the perspective of information system use as social business action. Therefore it is relevant to raise an important question at this point: How does actability contribute to these knowledge domains – does actability reinforce contemporary knowledge or simply reinvent what is already known? This section is devoted to an elaboration of this topic, starting with the concept of usability.

16.3.1 Actability and Usability

As indicated above, there are many definitions of the concept of usability. One definition that has been generally agreed on is the one by ISO 9241-11 (1998), which reflects the broad concept of usability: ‘the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use’.

Now it is time to take a closer look on how this definition relates to the definition of actability: an information system’s ability to perform actions, and to permit, promote and facilitate the performance of actions by users, both through the system and based on information from the system, in some business context.

By use of the term ‘a product’, ISO refers to both software and hardware. Such a product can be ‘used … to achieve specified goals’, that is, to produce intended outcomes. A user is anyone who interacts with the product. By ‘specified users’, ISO implies that different users might have different needs and comprehend the product differently. The terms ‘effectiveness’ and ‘efficiency’ imply that the specified goals are to be achieved with accuracy and completeness, and with as little expenditure of resources as possible. Furthermore, the goals should be achieved with ‘satisfaction’, that is, without discomfort and with positive attitudes towards the use of the product. The ‘speci-

---

fied context of use’ includes users, tasks, equipment, and the physical environment, where ‘task’ is defined as the ‘activities required to achieve a goal’.

From a usability perspective, the purpose of using an IS is to achieve specified goals. Actability, on the other hand, emphasizes the performance of social actions. Since actions are supposed to lead to desired business effects, these two formulations appear to be similar. A problem with a focus on goals is that it might lead to an overly instrumental view of human action, and thus of the performance of tasks. Actions are multifunctional and an action might lead to several business effects and several actions might lead to the same or a similar effect. In addition, actions are performed so as to conform to social norms. When designing for such a social understanding, a heavy focus on task goals might lead to decreased user satisfaction, if satisfaction is understood in a broader sense, with respect to the users’ social lives. What if a person, for example, feels misunderstood, not trusted or simply ‘impossible at work’? Such issues are hard to relate to achieving task goals in an instrumentally effective and efficient way. They rather relate to how one manages to create a mutual understanding with other people and act upon that understanding to achieve communicatively oriented goals.

A related issue is the use of ‘specified’ throughout the definition of usability. The set of actions that it is possible to perform through an IS is always ultimately derived from predefined rules, which are regulated during systems development. This is not reflected in the definition of actability, though. Finding out new ways of using an existing system might be a good thing. Nonetheless, it is important that users know what consequences their ‘improvements’ have for the action relationships created.

A special case of ‘specified’ in the definition of usability is the formulation ‘specified users’, which indicates that usability emphasizes user differences while actability assumes a typical user category. From this case, it is clear that usability is more strongly rooted in cognitive science than actability is. Even though actability builds on usability, the main focus of actability is not on cognitive aspects and user differences. Rather, actability tends to stress the importance of a balance between organizational and individual (cognitive) aspects of IS usage.

In general, the connotations of ‘user’ are different with respect to usability and actability. The usability focus is primarily on humans who directly and physically interact with a computer system. In the context of actability, a user is basically anyone affected by the actions performed through or by the system. More specifically, three meta-roles of users are identified: the performer who performs an action interactively, the communicator on whose behalf the action is to be performed, and the interpreter towards whom the action is directed.

It seems that the usability phrase ‘effectiveness, efficiency and satisfaction’ and the actability phrase ‘permit, promote and facilitate’ reflect similar aims. If a system permits and facilitates actions, it is probably effective and efficient as well. If it promotes action, it can probably be used with satisfaction. From an actability perspective it is important to interpret the usability terms from a social as well as from an instrumental orientation (see Chapter 7). Therefore the opposite implication is not necessarily applicable. Even if a system is effective and efficient it may still not permit, promote and facilitate actions, from a social action perspective. That is if those actions cannot be performed as successful communicative actions.

According to ISO 9241-11 (1998), usability should be understood within a ‘specified context of use’ – which tends towards the view of ‘one user using one computer’. This narrow human-computer view excludes other users who benefit from the IS. This
restriction has also been criticized by, for example, Schmidt and Bannon (1992). Their criticism is made from the perspective of Computer Supported Cooperative Work (CSCW), and Bannon (1991) claims that usability must widen the focus to ‘encompass groups of people and machines’. The actability concept of ‘business context’ suggests a specific context of use, which is a social context wherein people cooperate to do business by the use of information systems.

The usability concept of ‘product’ might correspond to the actability concept of ‘IS’. However, actability focuses its interest on information systems and is thus not primarily concerned with residual issues such as hardware configurations and workplace ergonomics.

A general observation is that the usability definition reflects more concern with measurability and has more of an engineering flavour compared to the more qualitatively and socially oriented actability.

So, does actability reinforce or reinvent usability? The answer is that ‘it depends’. It depends on what you mean by usability. If usability is understood in the narrow sense, as dealing only with user interface design and IS ease of use then yes, actability reinforces usability. The reinforcement consists of a solid grounding of the concept outside the realm of individuals’ interaction with artefacts; of taking the business context rather than the IS as point of departure for understanding IS use. If, on the other hand, usability is understood in the broad sense, as dealing with contextual studies of information systems in use, then the answer is not as simple. Usability in the broad sense and actability reflects similar aims. However, actability does not simply reinvent and repackage the broad concept of usability. Actability brings another perspective on information systems and their use within a business context. In this way it gives an opportunity to reinterpret the concept of usability. Actability points at issues not obviously brought to the foreground if usability principles are used unreflective. Actability is not usability and usability is not actability. Still, a truly usable system would be a system that is actable, and an actable system would certainly be usable. The important thing is to promote the qualities stressed by the actability perspective to achieve usable systems.

### 16.3.2 Actability and Communication Modelling

From the perspective of communication modelling (including language/action and semiotic perspectives), information systems are viewed as communication systems, as distinct from strict representational views of information. The strict representational view can be challenged, which the actability perspective certainly does. In the language/action perspective, information systems are not considered as ‘containers of facts’ or ‘instruments for information transmission’. Instead, the language/action perspective emphasizes what users do while communicating through an IS. Information systems are systems for business action, and business action is the means by which business relations are created. The aim of an IS is to permit, promote, and facilitate desired business actions. From an actability perspective, designing an IS means suggesting and establishing an action potential. An action potential both enables and delimits actions. It entails a repertoire of actions and a related vocabulary. The vocabulary consists of concepts related to the business language. An IS must also offer a record of actions performed. Information about these performed actions can normally be found in the IS database – the action memory of the IS.
The language/action perspective focuses on users performing actions. The meaning and purpose of acting is emphasized. The language/action perspective also discusses artefacts needed for performing tasks, but the emphasis is on theoretical frameworks for the analysis of human communication in organizational settings. This includes the performance of actions within a social business context. Language/action approaches do not generally include sufficient descriptions of the relationships between the actors and the artefacts used.

The best known approach within the language/action perspective is probably Action Workflow (Denning and Medina-Mora, 1995). Other important approaches are DEMO (Dietz, 1994; 2001), SAMPO (Auramäki et al., 1988), and Commodious (Holm, 1996). In these approaches business processes (with the support of information systems) are described as communication action of various kinds. The DEMO approach uses a predefined set of communication action types structured in a specified way to describe business processes, similar to Action Workflow. Actability does not build explicitly on any generic business model, even though it is related to the six-stage model for business interaction called the BAT model (Business Action Theory) (Goldkuhl and Röstlinger, 2002). The description of business processes following this six-stage model is much more liberal compared to how Action Workflow and DEMO use their respective stage models. The predefined structure of actions is not imposed on the business process descriptions in BAT in the same strict way as in these other approaches.

Within the field of organizational semiotics, the work of Stamper and Liu are probably the best known (e.g. Stamper, 1996; Liu, 2000; Stamper et al., 2000). This work uses the semiotic framework as a way to conceptualize information systems. This has been shown to integrate well with the concept of IS actability even though some conceptual distinctions are evident (see Chapter 14). Another central theme within organizational semiotics is the importance of understanding norms as a basis for human action, which goes well with the concept of IS actability (see Chapter 7) as well as with other language/action approaches (de Moor, 2002).

These other communication modelling based approaches do not, however, seem to go into the detail of requirements definition and IS evaluation taking usability issues into account, even though the Commodious (Holm, 1996) and SOMA (Graham, 1998) methods, as well as the work of Liu (2000) show ambitions in that direction. Dietz et al. (1998) also point this out to be an important research topic within the language/action perspective.

Andersen (2001), on the other hand, proposes a range of issues in which semiotic and communicative approaches can be used to enrich the understanding of human-computer interaction (HCI), and by extension information systems. Specifically he suggests that semiotics may be helpful in making HCI more coherent because of its inherent bridge between physical, psychic and social phenomena; exploiting insights from older media while at the same time identifying characteristic properties of the computer medium; and in situating interactive systems in a broader social context.

Another approach, based on Semiotic Engineering (de Souza, 1993; de Souza et al., 2001), which is closely related to the issues addressed by actability is that by Prates et al. (2000). Their approach suggests how to evaluate the communicability of user interfaces in parallel with software usability. In this view, IT-systems are regarded as one-shot higher-order semiotic messages sent from designers to users ‘about the range of messages users can exchange with systems in order to achieve certain effects’ (de Souza et al., 2001, p. 462). Communicability then is ‘the property of software that
efficiently and effectively conveys to users its underlying design intent and interactive principles’ (Prates et al., 2000, p. 32). The IT-system as message is, in itself, a performer of communication acts in the interaction between the user and the system (de Souza, 1993). Inherent in the approach is a focus on a ‘one-user-using-one-computer’ setting. The main interest is the communication between designers and users, and between users and system, in order to ‘let designers appreciate how well users are getting the intended messages across the interface and to identify communication breakdowns that may take place during interaction’ (Prates et al., 2000, p. 32). As a complement to focusing this ‘vertical’ communication between designers and users, the main focus in this dissertation is on the ‘horizontal’ communication acts performed by users directed towards other actors when using the system. Knowledge of this latter type of communication is essential for the design of IT-systems as active mediators in business communication.

Actability has its roots within the language/action perspective. This does not mean that actability relies on structured business action models such as Action Workflow (Denning and Medina-Mora, 1995) and the conversation for action schema (Winograd and Flores, 1987) or on typologies for classifying speech acts, such as those by Searle (1979) and others. Rather, actability is based on the central language/action thesis that using language is to perform action. This thesis has been integrated with ideas from other related knowledge domains such as Weber’s (1978) concept of social action, conceptual modelling (see Section 16.3.3) and usability (see Sections 16.2 and 16.3.1). The language/action perspective has continued to evolve since its inception in the late nineteen eighties with the seminal work by Winograd and Flores (1987)66. Research on actability represents one stream within that evolution. One important contribution to the language/action perspective made by actability is the ‘reintroduction’ of the propositional content in the language/action debate. In the tradition of the conversation for action schema, language/action approaches seems to have swung the pendulum too far in their critique of the descriptive approach. As a consequence, language/action approaches tend to focus too much on what speaking does, thus forgetting the important issue of what is spoken about. The actability approach to conceptual modelling is a way to remedy this misconception, and is an important contribution to reinforcing the language/action perspective. Another important contribution is the shift in focus from abstract business communication models towards theorization about the artefact per se.

16.3.3 Actability and Conceptual Modelling

One basic assumption within the field of, for example, entity-relationship modelling (Chen, 1976), and object orientation (e.g. Rumbaugh et al., 1991; Booch, 1994) is that the information system should serve as an image, or simulation, of reality. The information system could then be used to inform its users about the world, so that users do not have to observe the world directly. However, as recognized by actability, there are dimensions in our minds beyond the strictly conceptual one. People do not just look at the world and talk about it. They do things, act and communicate in the world. Utterances carry more than mere facts that tell something about something. They also carry the actor’s intention. Utterances per se can therefore be viewed as actions. These prop-

66 The ideas underlying the language/action perspective predates this work and goes back to research carried out independently by scholars such as Flores and Ludlow (1980) and Goldkuhl and Lyytinen (1982). However, it was the book by Winograd and Flores (1987) suggesting language/action as ‘a new foundation for design’ that boosted interest in the research area.
erties of information as action are seldom discussed or taken into account in traditional approaches to conceptual modelling.

The key concept in traditional dynamic modelling is the event (Jackson, 1983; Cameron, 1989; Coad and Yourdon, 1991; Rumbaugh et al., 1991; Booch, 1994; Rumbaugh et al., 1999). Booch (1994) defines event as ‘… some occurrence that may cause the state of a system to change’. However, in a social context, things most often do not just happen. An event, from the information system’s point of view, is almost always a result of some action performed by some actor. Since social action by definition is intentional, such an action is similar to an event but with additional attributes and constraints. The important thing is to identify the ‘events’ needed for system design based on the actions performed in the business. In that way the requirements specification does not just state what the system must be capable of doing but also why. Note that the business action model can also be used to identify human-IS interaction points (interactive use-situations and e-interactions). Traceability from computerized system to business is thereby straightforward as regards both functional requirements and sequence restrictions. Snoeck & Dedene (1998) remark that ‘modelling interaction by means of common event types requires the identification of relevant business event types’ but they do not say how to do it. From an actability perspective, such relevant business event types are to be found by identifying business actions. It is interesting to note that Blaha & Premerlani (1998) suggest that dynamic modelling most often is not necessary when developing database applications. They claim that objects in such a system do not have interesting dynamic behaviour. That is a claim in contradiction to Snoeck & Dedene (1998). The latter’s method M.E.R.O.DE. is mainly focused on dynamic modelling and the concept of shared events – similar to JSD (Jackson, 1983; Cameron, 1989). It is arguable that all actions, with corresponding events, that rely on or affect (create, modify or delete) the deontic state of the business, are important to capture in a conceptual model. This includes the notion that business actions themselves are important class (entity) candidates to consider.

16.3.4 Actability and Requirements Engineering

Much due to the widespread use of the Rational Unified Process (Kruchten, 1999), the use case concept has come to play a major role in requirements engineering (and throughout systems development). The use case concept (Jacobson et al., 1992; Jacobson et al., 1994; Jacobson et al., 1999; Booch et al., 1999; Rumbaugh et al., 1999) is an important inspiration for the actability concept of use-situation. Also the VIBA way of deriving interactive use-situations is quite similar to the way Jacobson et al. (1994) use the business object model to define use cases for the supporting information system. That similarity can be useful during systems design and implementation due to reuse of knowledge from the area of scenario-based systems design (Carroll, 1995). One major difference, however, lies in the fact that Jacobson’s et al. (1994) definition of use case comprises ‘…flow of events’ as a central notion. In contrast, the interactive use-situation of actability (see Chapter 4) is constituted by ‘flow of business actions’. Therefore the criticism of the concept of event in Section 16.3.3 is applicable also to the use case concept.

A problem regarding use cases is on which level they should be defined. Within actability, activities are aggregated based on the actors’ intentions with their constituent e-actions to delineate use-situations. That approach should also be applicable to use cases if a move from the concept of event to the concept of action was performed.
Information systems actability

Goal driven RE (Yu and Mylopoulos, 1998) is another area that is relevant to this work. Communication action is always based on some intentions and intention can be viewed as a kind of goal. Goals are, for example, used in VIBA and during actability evaluation when analysing the existing business. The goals of current actions and action structure can be used to transfer seemingly tacit knowledge and good practice when designing new businesses.

16.4 Lessons Learned and What to Do Next

Our chosen research approach implies that our knowledge of information systems actability at any given point in time (such as at the time of writing of this dissertation) is not the ultimate answer to the research questions – the concept will continue to evolve. This section summarizes the lessons learned so far, as reported in the previous parts of the dissertation, and elaborates on some possible ways to continue this work towards an even more grounded concept of information systems actability.

16.4.1 Actability Design Lessons

The lessons learned from working with the method VIBA for specifying information systems relate to (a) the development process, and (b) the forms of documentation (deliverables) suggested by the method. With respect to the development process, empirical experience clearly indicates that iterative development is preferable in order to manage changing requirements and communicate various models of the IS and its business context between developers (specifically between analysts/designers and programmers). Alternatively, one could have overlapping development teams with programmers participating also in the early workflows and analysts participating in the early construction steps. The development process model suggested, based on the RUP, is one way to facilitate such a setting. This approach has not been tried in any empirical setting with VIBA but seems to be a workable approach with the RUP (Kruchten, 1999).

One lesson learned regards the interplay between human-IS interaction modelling and business modelling. In order to get a complete understanding of the workings of the IS-supported business, a firm understanding of the IS itself has been shown to be important. Business modelling and user interface prototyping co-depend on each other and should be performed integrated. A related issue is avoiding making user interface prototypes too detailed. Thus, there is a need to balance the level of detail in prototypes with the level of detail in business models.

With respect to deliverables, the Actability Design research indicates that formalized specifications (such as visual models and tabular descriptions) are good for unambiguous communication but that they need to be complemented with verbal descriptions. An important question as yet unanswered with respect to VIBA is the required support from dedicated modelling tools. The UML parts of VIBA can make use of existing tools, such as Rational Rose, but there is a need for integrated tool support also for the non-UML parts.

The following lessons have been learned concerning the different VIBA deliverables:

- Action Diagrams are good for describing IS use within business processes contextually, but iterative tasks are hard to visualize.
The Use-Situation Lists proved themselves to be a reasonable way of connecting the different parts of the VIBA documentation, serving as a connection point between the business-focused documentation and the IS-focused documentation.

I-Tables give good support for implementing required functionality and connecting business requirements with detailed requirements on IS design. Still, there are issues concerning how to group e-interactions and corresponding I-Tables to better see how things relate. For trivial e-interactions, I-Tables are too detailed to be meaningful.

Document Prototypes have proven to be of great value, which has also been confirmed by other studies.

Document Definitions may be both unnecessary and too formalized to be used as an analytic tool. On the other hand, they may serve as an important source of design knowledge and rationale as part of a system’s specification for future use during system enhancements.

Document Statecharts are highly formalized descriptions of possible ways to manipulate the user interface and have proven to be a useful analytic tool once the learning threshold has been overcome. This is also acknowledged by Horrocks (1999).

Class Diagrams and Class Definitions are of course invaluable tools for conceptual modelling. Of course, another modelling language than UML can be used for the same purpose, but with respect to Learnability, the UML seems most promising due to its widespread use.

16.4.2 The Future of Actability Design

The future direction of Actability Design may take on (at least) two different forms.

The first would consist of further refinement and empirical tests of VIBA. This would include the complete specification, construction and implementation of information systems of various kinds. This way, the actual effects of the method could be evaluated based on constructed systems’ use in actual businesses.

The other form of future Actability Design research would consist of integrating the actability perspective in other existing methods, such as the RUP. This would mean integrating methods at the level of perspective, and the author of this dissertation has already initiated some work in this direction (Ägerfalk and Åhlgren, 1999; Ägerfalk and Wistrand, 2003). This way of approaching empirical Actability Design work may be more feasible since co-operating industrial partners do not need to use a completely different method, but enhance the one they are already familiar with.

Of course, a combination of both ‘in-house’ VIBA research and method integration research is possible, and a likely approach for further Actability Design work.

16.4.3 Actability Evaluation Lessons

The lessons learned from working with the Actability Evaluation method relate to (a) the suggested evaluation process model, and (b) the analytical tools used in the method: the nine actability principles, the D.EU.PS. model, and the layered model of action.

With respect to the suggested evaluation process model, the empirical experience indicates a need for flexible adoption. The two phases (ideal typical and situational) may have to be used integrated, while it is still possible to distinguish them conceptually. The reason for distinguishing them is that they call for different ways of working
and require different types and amounts of resources (for example, no users and less time during ideal typical evaluation).

The suggested actability principles have proven to direct attention to issues concerning communicative aspects such as responsibilities and living up to commitments made. Still, it may be the case that other principles are as important. In fact, the various existing usability principles naturally serve as an important complement. It is also evident that to apply the actability heuristics successfully, a thorough understanding of the underlying concept of actability is required.

The D.EU.PS. model, as an analytic tool, supported the evaluators in directing attention to aspects such as what was used by whom and why.

In conjunction with the layered model of action, the D.EU.PS. model may serve as an important tool to decide the level of detail in which to perform evaluation. This can be developed further and there is a need for support in the evaluation model to choose the level of evaluation.

16.4.4 The Future of Actability Evaluation

The development and refinement of the Actability Evaluation model will continue at least until the end of 2003, which is the time frame for the research project from which it emanates. The author of this dissertation and the other project members has put forth several ideas and suggestions for further development. From the empirical work, the following important issues have come up, which will be dealt with in the near future:

Â There is a need for integrating method support for adapting the evaluation process model. This is important in order to understand, for example, in what circumstances a shallow (ideal typical only) evaluation is sufficient. If time and resource constraints alone decide, then all analyses would probably end up being shallow. It is also important in order to decide how, and to what extent, it is feasible to iterate between the two overall phases (ideal typical and situational).

Â The method could probably benefit from a more elaborated integration of different scientific enquiry approaches. Today, the method makes use of qualitative evaluation methods based mainly on Patton (1990), Nielsen (1994) and Beyer and Holtzblatt (1998). These approaches could probably be combined with Grounded Theory like inductive approaches (e.g. Strauss and Corbin, 1998) to assist in revealing problems not easily captured at first. An interesting issue would be to incorporate more quantitatively oriented methods also.

Â There is a need for integrating method support for choosing appropriate level of analysis based on the layered model of action, depending on system and organizational setting. The layered model of action could also be expanded to fully take into account the 7+1 stages of e-actions and e-interaction (see Table 4-4, p. 72). This way an IS could be evaluated at the levels of use-situations, complete e-actions, seven (plus one) stages of e-actions, e-interaction, or seven (plus one) stages of e-interactions. Of course, such increased detail and flexibility would demand even more elaborate method support for choosing appropriate level.

Â More research on the interplay between actability evaluation and usability evaluation is required. One way to approach this issue would be to perform further comparative studies using existing usability principles (such as Nielsen’s (1994) heuristics) and the suggested actability principles. This would also include the exploration of how much knowledge of the theoretical basis of actability is required to perform actability evaluation. Is it possible to operationalize the heuristics even
further so that non-experts in actability can perform actability evaluation with satisfactory results?

Another, complementary way to continue, and to broaden, the work on Actability Evaluation, would be to investigate the relation between actability evaluation and general IS evaluation approaches more thoroughly, focusing more on IS success, in the wake of, for example, DeLone and McLean (1992) and thus on, for example, cost/benefit analyses in economic terms. This is, however, outside the scope of the ongoing project and is therefore to be considered possible future research.

16.4.5 Actability Studies Lessons

The lessons learned from the two Actability Studies relate to the analytic descriptive framework that was used in the studies.

In the first study, concerning Internet-based information systems, the analytic framework helped to identify differences in emphasis between Internet-based systems and traditional systems, and to show that the characteristics themselves are not unique. The work shows that actability can serve as an effective tool in directing attention to communicatively oriented aspects of information systems. In this first study, actability helped to isolate the different types of actors involved in the Internet-based systems context. This understanding then served as the baseline for the understanding of characteristics of Internet-based systems, such as the concept of ‘degree of interaction’ and the relation between ‘power over’ and ‘knowledge about’ users. In addition to actability, the semiotic framework was used as an important inspiration. That way, an analytic framework that balances the human aspects of information systems with the more technological aspects was achieved. The actability concept contributed with a conceptualization of actors, artefacts, actions and their relations. The stricter hierarchical partitioning imposed by the semiotic framework brought actability even further. When it comes to the artefact, the semiotic framework also helped in focusing on different aspects more independent from each other. Nonetheless, applying actability in accordance with the semiotic framework was not completely straightforward. There are important conceptual and terminological differences. Another occasional problem was the complex mapping of empirical categories to distinct semiotic levels. It proved to be the case that some categories could not be related to one single level without separating them into possible sub-components. These problems led to a continuous redefinition of the categorization based on the analytic framework, such as the relations between trust, information security, and validity. Altogether, the analytic framework proved useful in finding and relating different important categories of Internet-based systems.

In the second study, concerning the Local Electronic Marketplace (LEMP), the analytic framework was extended with an explicit and comprehensive notion of practice, adopted from Goldkuhl and Röstlinger (1999). The results show that there is more to gain in terms of synergy between the digital and physical channels related to the LEMP if these are viewed as tools for performing social action. The benefit of using actability and the notion of practice as a base for understanding this multi-channel interplay is that it helps to identify the LEMP as a social practice spanning company borders, and provides guidance as to how to design information systems that truly support such a practice.
16.4.6 Future Actability Studies

No future actability studies are planned at the time of writing of this dissertation. In our future work on method configuration (see Ågerfalk et al., 2003), there may be opportunities for using the actability analytic framework as an aid in the identification of different types of systems and development situations. This was also one of the reasons for carrying out the Internet-based systems study in the first place (see Karlsson et al., 2001). Of course, it is the author’s wish that what is presented in this dissertation will constitute incentives for other people performing IS field work to adopt, if not the whole analytic framework, at least some of the ideas presented.

16.4.7 Actability on Reflection

16.4.7.1 The Applicability of Actability

In this dissertation, actability has been discussed and applied within certain types of systems and business contexts. More specifically, the concept and its operationalizations have been applied in production planning in mechanical engineering, material administration in paper production, resource management in education and manufacturing, home care planning, electronic commerce, and Internet banking. The types of systems involved in the work have been traditional, client/server type of systems as well as Internet-based systems, including local electronic marketplaces.

This indicates that the concept is applicable in various different types of business contexts and systems. This also means that we cannot really say anything about the applicability of actability outside of these domains. However, there are theoretically justified reasons to believe that it would be relevant to use the concept also in other domains in which information systems are used for communication and social action.

Certainly, future work on actability should strive to include other domains in order for actability to evolve, and adapt to different settings.

16.4.7.2 Operationalization of a Concept

In this dissertation, actability has been operationalized into method support for systems development and evaluation, and analytic support for qualitative IS studies. Certainly, other forms of operationalization could be made in future research. One such operationalization would be to use an actability framework in method engineering, and specifically method configuration, as suggested above (see Section 16.4.6).

16.4.7.3 Expanding the Scope of Actability

Actability, as it has been discussed in this dissertation, has been restricted to the use of information systems within different business contexts – the discussions have been restricted to ‘information systems’ actability. It may very well be that the concept is applicable also to the use of other types of artefacts and organizational assets and abilities, and the underlying theoretical base certainly is (as evidenced by a good deal of the research on the language/action perspective). Initial work on broadening the scope of actability in this direction has been performed and reported by the author’s colleagues Braf, Goldkuhl and Röstlinger (Braf and Goldkuhl, 2002; Goldkuhl and Röstlinger, 2002; Goldkuhl and Braf, 2002)

Another area for which actability seems appropriate is that of multi-agent systems. Technical issues have dominated this evolving area so far, and much remains to be done from a business and social action point of view. Important research in this area has been carried out by Liu and colleagues (Liu et al., 2001b; Chong and Liu, 2002)
based on organizational semiotics. As shown in this dissertation, the theoretical basis provided by organizational semiotics can be integrated with the concept of actability, and could be a fruitful way of approaching multi-agent systems from a business and social action perspective.

### 16.5 Summing up the Summary

The perspective held by actors influences their actions. Different perspectives emphasize different aspects, and several perspectives may be applied during IS work. Actability promotes a particular perspective that emphasizes pragmatic aspects of information systems’ business use. By externalizing actability, these aspects may be consciously and reflectively considered in systems work, such as during the development and evaluation of information systems. You may not like actability. You may not believe that actability is important. You may not like the way the validity of actability has been argued in this dissertation. However, by presenting this dissertation, the author hopes that IS people will at least consider the actability construct. More important, by presenting the concept of actability and some of its major applications, it becomes possible to discuss whether it is worth bothering with or not; whether special measures are needed for it to be effective in information systems or whether the fact that actability properties may eventually emerge by chance will suffice.

In this dissertation, IS actability has been described as a theory and a perspective that identifies IS usage quality with an IS’s potential of permitting, promoting and facilitating business action. This means that understanding the use context of an IS becomes, not only important, but actually imperative for IS success. Understanding the IS use context as a business context consisting of actors, actions, goals, intentions, and information systems, gives an opportunity to minimize the potential gaps between business requirements and system requirements, and between organizational issues and individual (cognitive) issues. Understanding that information systems per se may take action within a business gives us an opportunity to understand responsibilities related to IS development and use as responsibility for actions performed, and thus interpersonal action relationships created and maintained. Focusing on the use context from a pragmatic perspective means regarding pragmatic issues, such as action responsibilities, not only as a result of IS design, but as its point of departure. Viewing information systems as information action systems makes it possible not to get trapped in a reductionist view of information systems as ‘containers of facts’ but as active tools and mediators in ongoing business conversations.

### 16.6 Some Final Words

This dissertation is concerned with theory and method for information systems work. The work reported has been carried out as a combination of theoretical work and empirical research. ‘Sure it works in practice, but will it work in theory?’ The question was brought up by Erik Clemons during his keynote speech at the 1999 IRMA conference in Hershey Pennsylvania – a question with humorous undertones but with serious implications. One aim of the research reported in this dissertation is to create (or reveal, depending on metaphysical preferences) knowledge that ‘works’ both in theory and practice. To theorize about practice and to make theory practical is, according to this author’s belief, the task with a capital T of academic IS research. Unfortunately it sometimes seems easier to try to change good practice than to change bad theory.
Index

A
actability ................................ 4, 54, 64
actability principles ............... 210, 232
action ........................................ 38, 57, 250
Action Diagram ...................... 159, 186
action elementariness............... 211
action elementary message ...... See ae-message
action knowledge...................... 13, 38
Action Modelling...................... 162
action potential......................... 6
Action Set .................................. 175
action space.............................. 6, 219, 268
Action Workflow ..................... 106
Activity Set .............................. 174
actor ......................................... 38, 57, 67
ae-message ............................... 64, 88, 165
application................................. 41
approach ..................................... 44
artefact ...................................... 38, 62
ASP .............................................. 162
automatic action......................... 63
automatic use-situation ............. 65, 163

B
Business Analyst......................... 150
Business Definition..................... 156
business process....................... 129, 192

C
Class Definition .................... 172, 189
Class Diagram......................... 172, 189
Class Statechart .................... 172
client configuration.................. 259
cognitive approach..................... 9, 279
cognitive base........................... 38
communication act..................... 54, 73
communication modelling........... 46, 282
communicator ............................ 67
concept ...................................... 41
conceptual consistency .............. 172
conceptual modelling............... 101, 171, 284
Conceptual Set ......................... 176
consequence .............................. 41
consequential action .................. 63
consequential use-situation ....... 65, 163
content supplier ......................... 268
context ...................................... 118, 278
contextual approach ................. 5
contextual enquiry ..................... 209
contextualization ....................... 5, 278
conversation-for-action ................ 105
CSP ............................................ 162

d
D.EU.PS ................................. 222, 236
degree of interaction ................. 258
delayed feedback ....................... 214
delayed interpretation ............... 213
delegated action ......................... 214
deliverable .................................. 150
deployment architecture .............. 258
descriptive fallacy ...................... 12
development process model ....... 28, 154
document .................................... 71, 166, 169
Document Definition .............. 166, 169, 188
Document Prototype ................ 166, 169, 187
Document Statechart ................ 171, 188

e
e-action ...................................... 64
effectiveness ............................. 6, 122
efficiency ................................. 6, 122
e-interaction .............................. 68, 167
e-Interaction List ...................... 167
elementary action ....................... See e-action
elementary interaction See e-interaction
e-message ................................. 63, 83
der end user ................................. 150
evaluation process model .......... 207, 231

F
focal area ................................... 147

G
generic target groups ................. 255
| **Glossary** | 156 |
| **goal** | 157 |
| **Goal Diagram** | 157 |
| **Goal List** | 157 |
| **goals-based evaluation** | 207 |

### H

| **habitant** | 268 |
| **heuristic evaluation** | 207, 235 |
| **host** | See host organization |
| **host organization** | 255, 268 |
| **hybrid approach** | 263 |

### I

| **ideal typical analysis** | 208 |
| **illocutionary act** | 54 |
| **in-context research** | 36 |
| **individual approach** | See cognitive approach |
| **infological equation** | 84 |
| **informatics** | 21 |
| **information security** | 257 |
| **information system** | 62, 65 |
| **input message** | 96 |
| **instrumental act** | 117 |
| **instrumental fallacy** | 11 |
| **instrumental goal** | 119 |
| **Interaction Modelling** | 167 |
| **Interaction Set** | 175 |
| **Interaction Table** | See I-Table |
| **interactive action** | 62 |
| **interactive use-situation** | 65, 163 |
| **internal quality** | 4 |
| **internals** | 255 |
| **Internet-based system** | 254 |
| **interpreter** | 67 |
| **interpretivism** | 36 |
| **inter-subjective world** | 38 |
| **irrevocable action** | 213 |
| **ISP** | 146, 162 |
| **I-Table** | 167, 186 |

### J

| **JSD** | 285 |

### K

| **knowledge** | 40 |

### L

| **language/action perspective** | 23, 282 |
| **LEMP** | 263, 271 |
| **local electronic marketplace** | See LEMP |

### M

| **marketplace functionality** | 267 |
| **message** | 55 |
| **Message Definition** | 165 |
| **method** | 24, 33, 34 |
| **method component** | 28 |
| **method fragment** | 28 |
| **methodology** | 25 |
| **multi-agent systems** | 290 |
| **multi-functionality** | 87 |

### N

| **network configuration** | 259 |
| **norms** | 59, 117 |

### O

| **object orientation** | 130 |
| **ontology** | 38 |
| **operationalization** | 41 |
| **organizational action** | 61 |
| **organizational approach** | 9 |
| **organizational semiotics** | 46, 54, 283 |
| **output message** | 96 |

### P

| **paradigm** | 23 |
| **partners** | 255 |
| **performer** | 67 |
| **perlocutionary act** | 54 |
| **perspective** | 22, 27, 275 |
| **positivism** | 36 |
| **practice** | 44, 264, 265, 291 |
| **pragmatism** | 38 |
| **pragmatization** | 10, 279 |
| **Prerequisite Set** | 173 |
| **prerequisites** | 156 |
| **problem** | 157 |
| **Problem Diagram** | 157 |
| **Problem List** | 157 |
| **Process Modelling** | 155 |
| **propositional act** | 54 |
| **public** | 255 |
Index

R

rapid development ............. 129, 193
recorded action.................. 212
remote interpretation............ 213
requirements engineering........ 46, 285
requirements management....... 131, 195
Role Definition................... 162
Role Diagram..................... 162

S

satisfaction..................... 6, 122
semiotic framework.............. 54, 249
semiotic ladder................. See semiotic framework
server configuration............. 259
sign............................. 38, 55, 78, 84, 249
situational analysis............ 209
social action..................... 3, 87
social goal........................ 119
software quality................ 4
speech act........ See communication act
Statechart........................ 170
structured action................ 213
subjective world................ 38
Supplementary Prerequisites..... 158
system owner..................... 150
systems analyst................ 150

t T

task................................... 74
technical security................ 259
time to deliver................... 260
tradition.......................... 23

U

unit of analysis............. 45, 135, 201
usability...................... 5, 116, 131, 194, 280
usability principles........... 216
usage fallacy................... 10
usage-centred................... 277
use case............................ 74, 165
usefulness....................... 7
user.................................. 10, 67
user interface designer......... 150
use-situation.................... 65, 162
Use-Situation Diagram......... 164
Use-Situation List............. 163, 186
utterance act.................... 54

V

validation......................... 42
validity......................... 49
VIBA.................................. 146
vision.............................. 158
Vision Document................ 158
visitor........................... 268

W

worker......................... 150
workflow......................... 149
References


Ågerfalk P J (1999a) Encapsulation or Availability: On the Combination of Objects and Rela-
Seminar in Scandinavia (IRIS 22), 7–10 August 1999, Keuruu, Finland, Vol. 1, (Ed, Käkölä

Outline, Licentiate Thesis, Dept. of Computer and Information Science, Linköping Univer-
sity, Linköping.

104.

Ågerfalk P J (2002) Messages Are Signs of Action: From Langefors to Speech Acts and Beyond,
In Proceedings of the 7th International Workshop on the Language-Action Perspective on
Communication Modelling (LAP 2002), 12–13 June 2002, Delft. The Netherlands, (Eds,

Ågerfalk P J and Åhlgren K (1999) Modelling the Rationale of Methods, In Managing Informa-
tion Technology Resources in Organizations in the Next Millennium, Proceedings of the
10th Information Resources Management Association International Conference, 16–19 May
190.

Information Systems Use, Accepted for the 10th International Conference on Human-

Systems and the Future of the Digital Economy, Proceedings of the Xth European Confer-
ence on Information Systems (ECIS 2002), 6–8 June 2002, Gdansk, Poland, Vol. 1, (Ed,

Ågerfalk P J and Eriksson O (2003) Usability in Social Action: Reinterpreting Effectiveness,
Efficiency and Satisfaction, Accepted for the 11th European Conference on Information Sys-

the New Millennium, In Information Modeling in the New Millennium, (Eds, Rossi M and

Integrating Analysis of Business Processes and Usability Requirements, In Proceedings of
the 4th International Workshop on the Language Action Perspective on Communication
Modelling (LAP’99), 12–13 September 1999, Copenhagen, Denmark, (Eds, Goldkuhl G, et
al.) Jönköping: Jönköping International Business School, pp. 73–86.

Material), Örebro, Sweden: Dept. of Informatics (ESI), Örebro University.


Bühler K (1934) Sprachtheorie, Jena: Fischer.


References


Information systems actability


Mathiassen L (1982) Systemudvikling Og Systemudviklingsmetode, Doctoral Dissertation in Danish, Datalogisk afdeling, matematisk institut, Aarhus universitet, Århus, Denmark.


Information systems actability


Men kontrollen blinkar blå…
Department of Computer and Information Science
Linköpings universitet

Dissertations

Linköping Studies in Science and Technology


Linköping Studies in Information Science


