Learning and Using Methodologies in Information Systems Analysis and Design

Nielsen, Peter A.

Publication date:
1990

Document Version
Accepted author manuscript, peer reviewed version

Learning and Using Methodologies in Information Systems Analysis and Design

Information Systems Group
Dept. of Mathematics and Computer Science
Institute of Electronic Systems
Aalborg University
Denmark

Learning and Using Methodologies in Information Systems Analysis and Design

by

Peter Axel Nielsen
Cand.scient., Computer Science, Århus University

Submitted for fulfilment of the degree of Ph.D. in
Systems and Information Management, Lancaster University

August 1990

Supervisor: Dr. Brian Wilson

Internal Examiner: Dr. Ron Anderton

External Examiner: Prof. Robert Galliers

Contents

Abstract **v**

Preface **vii**

Introduction **1**

I Background **5**

1 Topic and Approach **9**
1.1 The Topic 9
1.2 Research Approach 13

2 Methodologies in Information Systems Analysis and Design **19**
2.1 Information Systems Methodologies 19
2.2 Information Systems Development 27
2.3 The Concept of Information Systems Methodology 32
2.4 Summary 38

3 Related Research **41**
3.1 Appreciation Approaches 42
3.2 From General Approaches to Situational Approaches . . 52
3.3 Summary 57

II Practice **59**

4 The Individual Level **63**
4.1 The Inquiry 63
4.2 Three Systems Developers 67
4.3 The Diversity of Individual Use 71
4.4 Summary 74

5 The Project Level **75**
5.1 The Inquiry 75
5.2 Following A Single Methodology 80
5.3 Theoretical Support 89
5.4 Combining Methodologies 97
5.5 Methodologies at the Project Level 114
5.6 Summary 118

6 The Organisational Level **121**
6.1 The Inquiry 121
6.2 The Course of Action 125
6.3 Learning of New Methodologies 130
6.4 Methodological Choice 136
6.5 Summary 140

III Implications **141**

7 Criteria for Using Information Systems Methodologies **145**
7.1 Introduction 146
7.2 Domain of Use (α) 148
7.3 Conditions for Use (β) 153
7.4 Characteristics of Use (γ) 159
7.5 Summary 165

8 Towards a New Approach **167**
8.1 The Ideas Behind the Approach 167
8.2 The Approach in Detail 170
8.3 Summary 178

IV Conclusion **179**

9 Thesis Conclusions **183**
9.1 Contributions to the Field 183
9.2 Soundness of The Research 187

A The Lessons Learned **191**

Bibliography **197**

List of Figures

1 The parts of the thesis 2
2 A picture of the content of the parts and chapters 3

1.1 The learning cycle of action research. 14

2.1 A system of information systems development 28
2.2 The relation between situation, working practice, and methodologies 35

3.1 A conceptual model used in the survey 43
3.2 The mapping of the appreciation approaches in summary 53
3.3 Four categories of appreciation approaches. 54
3.4 Significant differences between the two view-points on professional practice. 55

4.1 Case 1: The individual level. 64

5.1 Case 2: The project level. 77
5.2 Using diaries as a medium for evaluating activities 79
5.3 Using diaries as a medium for planning activities 80
5.4 One of the data-flow models made. 83
5.5 One of the partaker models used 92
5.6 Part of one of the transaction cost models used 93
5.7 Part of one of the process models used. 100
5.8 Part of one of the task models used 101
5.9 The initial model used. 103
5.10 Part of one of the modified process models used. 104
5.11 Two entity specifications 106

6.1 The methodological learning cycle 123
6.2 Methodological support and advice 126
6.3 Rich picture of the Provident 127
6.4 A system to produce an Information Architecture 131
6.5 A system to adapt methodological repertoire 138

7.1 Criteria for using information systems methodologies. . . 147

8.1 A new approach in outline 169
8.2 The activities of the new approach 170
8.3 Use-Criteria are mapped to conceptual models 173
8.4 Methodologies are mapped to the Use-Criteria 175
8.5 Deciding which methodologies or parts hereof to use . . . 176

9.1 The conceptual model used in the survey in Chapter 3 . 185
9.2 The activities of the New Approach 186

Abstract

This thesis addresses the fundamental question: 'Which methodologies in which situations?' Action Research has served as the primary research approach, i.e. the thesis is based on documented and practical experience. The argumentation throughout the thesis is rooted in this practice; the lessons learned, the interpretation and the reflection refer explicitly hereto or to the theoretical framework of the thesis. The thesis refers to three levels of practice:

- The individual level: Based on qualitative interviews with three systems developers lessons have been learned on the richness and diversity of individual use of methodologies.
- The project level: Based on the use of methodologies in three similar projects (documented in project diaries) several lessons have been learned. Some of the lessons address the question of the domain of usefulness of the involved methodologies. The remaining lessons address the question of how methodologies may be use: using only a single methodology, using a theory as a methodology, and using a combination of methodologies.
- The organisational level: Based on an inquiry structured by Soft Systems Methodology lessons have been learned on: the process of introducing new methodologies a DP department and the process of choosing methodology in a particular situation.

Each of the lessons learned are argued with explicit reference to the interviews, the diaries, or the soft systems models.

The implications drawn from this practice as to how to answer the fundamental question is twofold. On the one hand, it is possible and useful to have a framework to explicate differences between domains where different methodologies are useful. This has been done as a set of Use-Criteria related to: context of use, conditions for use, and characteristics of use. On the other hand, such a framework has itself limited usefulness as it is stable and general. A new dynamic and situational approach for learning and using methodologies is outlined. The approach is based on Soft Systems Methodology and integrates the Use-Criteria.

Preface

The research reported in this thesis is my attempt to understand the learning and use of methodologies in information systems analysis and design both at a *theoretical* and at a *practical* level. The intention has been to gain insight into the world of methodologies in information systems analysis and design by looking at the practice of learning and using such methodologies and — based on this insight — to develop a set of ideas, concepts, and guidelines for thinking and acting in this world.

The idea to make sense of methodologies goes back to my graduate studies in Computer Science at Århus University in 1984 to 1986. Here I took part in a series of courses where we studied different methodologies in information systems analysis and design. As an intrinsic part of these courses we experimented with the methodologies in a large bank in order to view them from a practical standpoint.

I was accepted as a Ph.D. student at the Department of Systems and Information Management at Lancaster University during Summer of 1986. My topic was at that time: 'Classification of Techniques in Analysis and Design.' Later, when I came to Lancaster in the Autumn of 1986 I met a new and exciting world of soft systems thinking. The contrast between the 'hard' and the 'soft' methodologies came somewhat as a surprise to me. It became clear that my initial topic was to some extent based on 'hard' ideas. The ideas behind my research then changed to a 'soft' set of ideas.

Returning to Denmark from Lancaster during the Summer of 1987 I began working at Department of Mathematics and Computer Science at Aalborg University. This provided me with an opportunity for doing the remaining practical work in a familiar language and culture. From Autumn of 1987 till the Summer of 1988 I did a research project concerned with methodological learning in a medium-sized DP department.

I have been forced by these circumstances to find my own way of bridging or coming to terms with the gap between Scandinavian and Lancastrian research traditions, the gap between hard and soft, the gap between theory and practice, and the gap between relevance and rigour in research. The challenge involved in this led me through a useful learning process questioning and clarifying my research. On the other

hand, I may not have fully succeeded in meeting the challenge, but I shall leave that for the reader to judge.

I am very much in debt to my supervisor, Dr. Brian Wilson, who has helped me in many ways during the last years with this thesis by encouragement, ideas, criticism, and patience. He and his colleagues at the Department of Systems and Information Management have by conveying their enthusiasm, their way of doing research, and by their achievements in soft systems methodologies changed my intellectual belief and desire more than I could ever hope for.

The Lancaster experience did not only force me to express my ideas and research in English it also exposed me to the professional 'soft systems' language at the department in Lancaster. I had the privilege to receive many invaluable comments on earlier drafts of the thesis from Keith Sawyer at Bristol Polytechnic. He did the huge and almost impossible work on correcting my language mistakes. Gitte Carøe has done a tremendous job in proof reading the final version. I must take the blame for the errors that remain.

My colleagues at the Department of Mathematics and Computer Science at Aalborg University have supported me by an endless stream of encouragement and discussion. Prof. Lars Mathiassen, my mentor in many respects, and Dr. Jan Stage have been kind enough to discuss the overall structure and the line of argumentation with me at several occasions.

I am grateful to those with whom I have done the empirical work reported in Chapter 5: S. Bengtson, P. Huber, O. Jensen, L. O. Jepsen, K. B. Munkholm, and K. Steffensen.

I am also grateful to the two organisations that provided us with an opportunity to intervene into practice (Chapters 4, 5, and 6). The organisations are anonymous in this thesis, but the difficulty they exposed us to and the resources they used have been invaluable to me. The Department of Mathematics and Computer Science at Aalborg University provided me with the facilities for writing the thesis. So did on occasions the Department of Computer Science at Århus University.

Some of the research behind this thesis has been reported before. The contents of Chapter 3 was initially an article in *Scandinavian Journal of Information Systems* and an earlier version appeared in *Proceedings of the 12th Information Systems Research Seminar in Scandinavia*. The part on diaries as a research approach in Chapter 5 has been fully reported in my Masters Thesis together with L. O. Jepsen and later in an article in *Behaviour and Information Technology* together with L. O. Jepsen and L. Mathiassen. The part on criteria for using methodologies

in Chapter 7 was initially written as a conference paper to appear in *Proceedings of the 10th Information Systems Research Seminar in Scandinavia* and later as an article in *Office Technology and People*.[1]

The research presented in this thesis has been financially supported by the Danish Natural Science Research Council, Grant No. 11-6079, and by Royal Society (U.K.), Grant No. 621003.F608/UR.

[1]The references are: Chapter 3: (Nielsen 1989c; Nielsen 1990), Diaries: (Jepsen and Nielsen 1986; Jepsen *et al.* 1989), Criteria (used to be 'distinctions'): (Nielsen 1987b; Nielsen 1989b).

Introduction

This thesis is about a research effort into the topic:

> *The learning and use of methodologies in information systems analysis and design.*

Methodologies play an important role in analysis and design of information systems and there are a great number of methodologies available. In practice, it makes a difference which methodologies are used, what they are used for, and by whom they are used. It also matters how well methodologies are learned and how they are learned. Moreover, there is a two-way street of learning and use: In order to use a methodology it must be learned and in order to learn a methodology it must be used. It is because of concerns and issues like these that research reported in this thesis has been done.

If there is a simple question that catches the totality of this thesis it is:

> *Which methodologies in which situations?*

The question summarises to a great extent what the research has been trying to answer. In dealing with this question action research has been applied. This means that reference to practice and reflection upon practice play a significant role in the thesis.

The thesis is structured in four parts: (1) background, (2) practice, (3) implications, and (4) conclusion, where Part 2 on the practice of learning and using methodologies is the most essential part. Figure 1 shows how the thesis is structured into parts. Part 1 presents the intellectual background for the research. This background is given in order to establish a platform from which the practice of analysis and design can be understood. Part 2 provides the lessons learned from three cases of learning and use of methodologies. Part 3 presents a number of implications based on the lessons of Part 2. Part 4 ends the thesis by giving the concluding remarks.

Figure 2 illustrates the structure of the thesis in terms of parts and chapters. The figure should be viewed in conjunction with the following description.

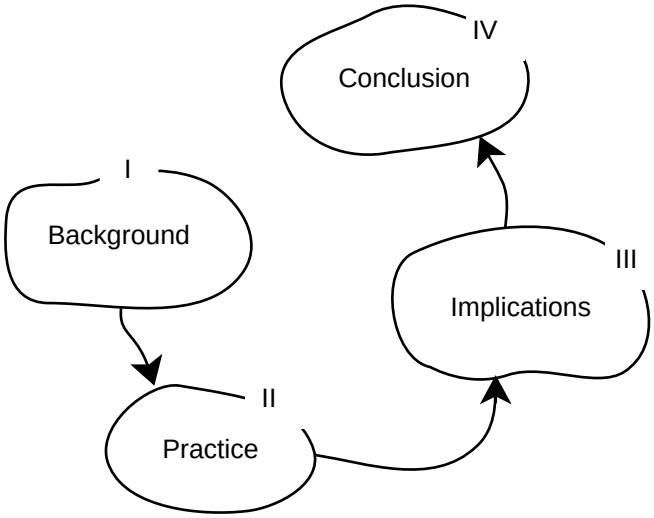


Figure 1: *The parts of the thesis. Key: ◯: a part; →: depends on the previous part.*

Part 1: Background

This part presents the intellectual background for the research in three chapters.

Chapter 1 outlines the topic by presenting the assumptions on which the research is based together with the research issues that are derivable from these assumptions. It also describes the research approach that has been applied.

Chapter 2 introduces the concept of information systems methodology. A brief presentation of some of the methodologies that are available is given, and the stance taken towards information systems analysis and design is presented as a systems model. The chapter ends with a discussion and definition of the concept of methodology.

Chapter 3 presents a survey of the literature on prior research into the subject of this thesis. This survey is conducted as a soft systems study where a number of the approaches found in the literature is mapped onto a systems model. Based on this, a more theoretical discussion advocates a move from approaches providing general answers towards approaches that can be used to develop unique answers in specific situations.

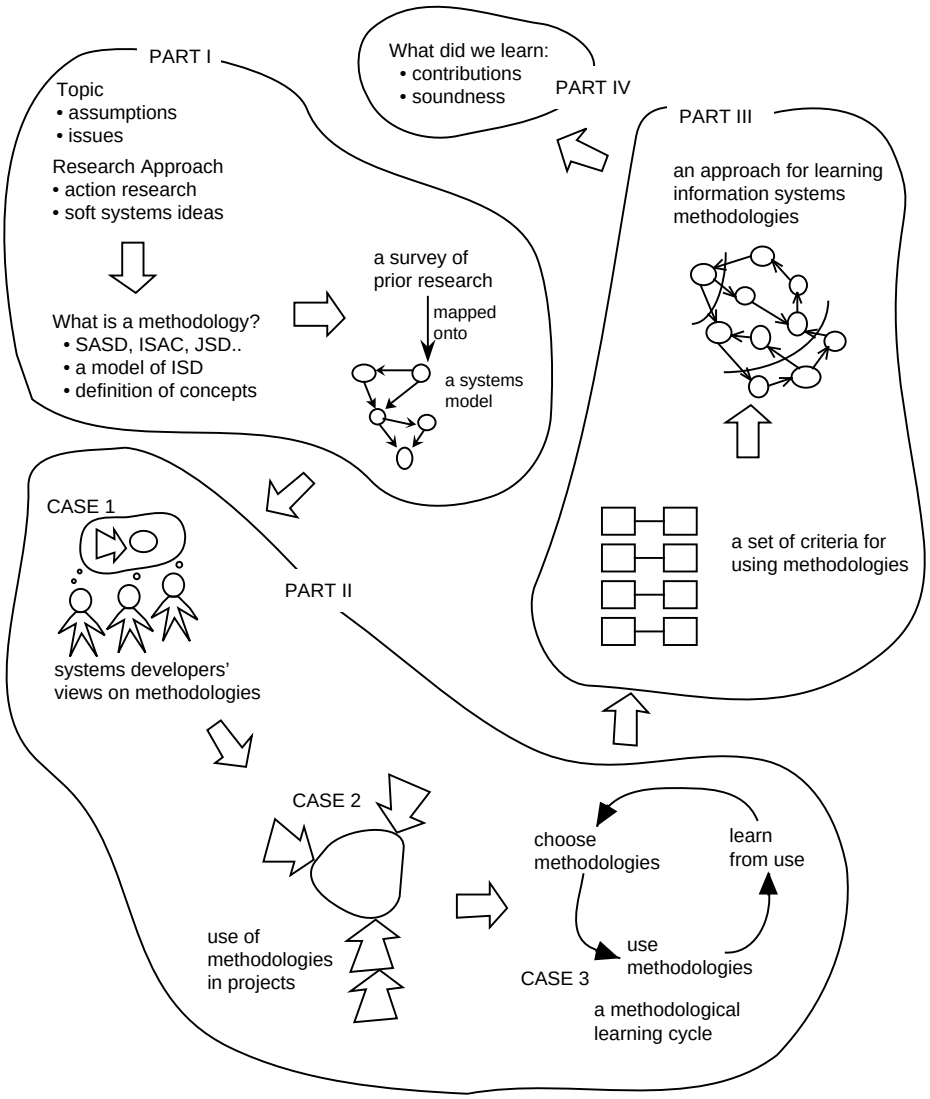


Figure 2. *A picture of the content of the parts and chapters as an expansion of Figure 1. Each chapter is depicted pictorially and the white arrows show how the story moves from one chapter to another.*

Part 2: Practice

This part reports and reflects upon three cases of practice. The cases are at three different levels of organisation and the lessons learned are different but related.

Chapter 4 is based on interviews with systems developers in a DP department. The lessons that have been learned from Case 1 are about systems developers' views and attitudes towards the use of methodologies. Case 1 represents research at the individual level.

Chapter 5 presents research at the project level. Case 2 consists of three projects each using different methodologies in similar situations. The lessons learned from this are about the usefulness of the specific methodologies and about the usefulness of different ways of using methodologies.

Chapter 6 is based on a soft systems enquiry into the introduction and learning of methodologies in a DP department. The lessons learned from Case 3 are about the fundamentals of the learning and use of methodologies at the organisational level.

Part 3: Implications

This part generalises and combines the lessons reported in Part 2 in two chapters.

Chapter 7 draws upon the lessons learned about the specific methodologies and formulates a set of criteria for using methodologies. The criteria are formulated in a way that highlight which decisions that have to be taken in a specific situation with respect to which methodologies to use.

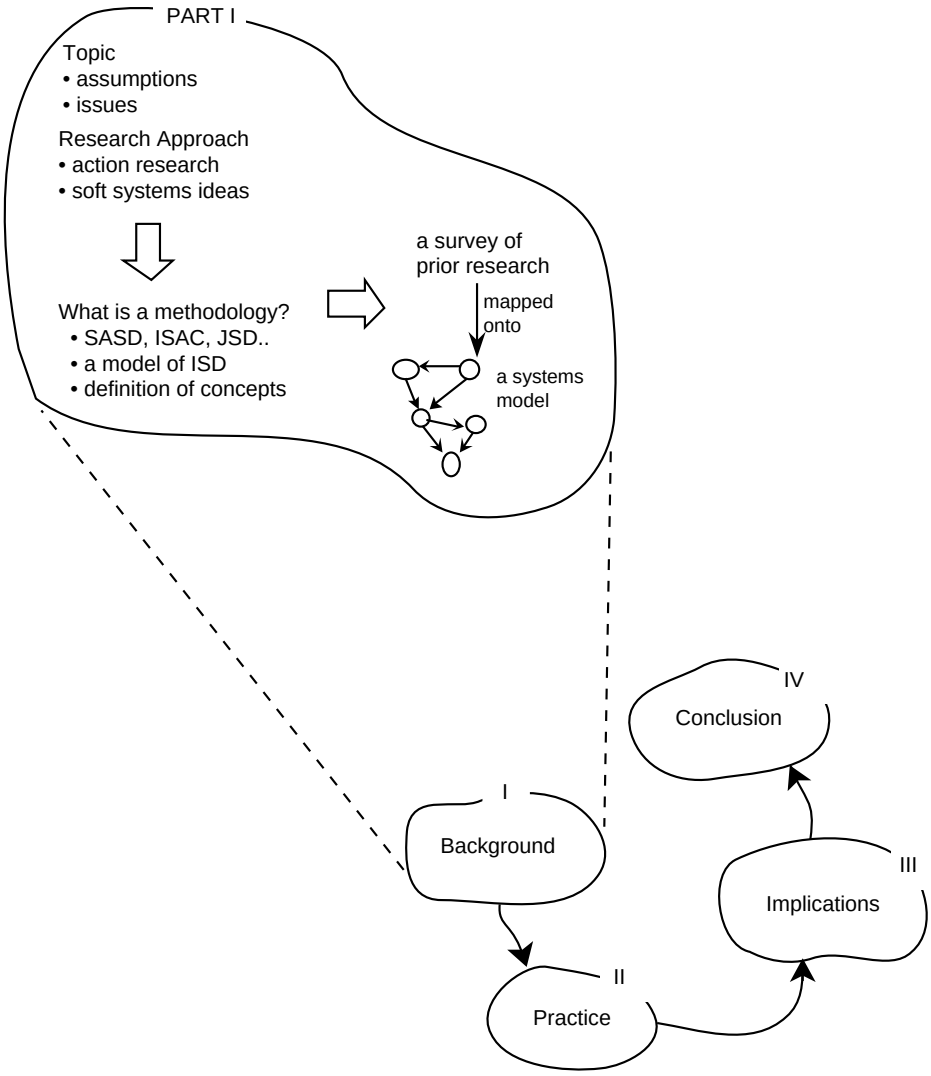
Chapter 8 outlines a new approach to be applied in practical situations for learning methodologies as an integrated part of using them. The approach is based on soft systems ideas and utilises the lessons learned and the criteria for using methodologies.

Part 4: Conclusion

This part concludes the thesis.

Chapter 9 discusses how the outcomes of this research contribute to the field. It also discusses the validity of the research.

Part I
Background


PART I
Topic
• assumptions
• issues
Research Approach
• action research
• soft systems ideas
What is a methodology?
• SASD, ISAC, JSD..
• a model of ISD
• definition of concepts
a survey of
prior research
mapped
onto
a systems
model
IV
Conclusion
I
Background
III
Implications
II
Practice

1

Topic and Approach

This chapter outlines the topic and approach of this thesis. Section 1.1 presents a set of assumptions and issues about methodologies in information systems analysis and design. Section 1.2 presents the research approach that has been applied to deal with the chosen topic.

1.1 The Topic

What to do and how to do it have always been crucial elements of analysis and design of information systems. These 'whats' and 'hows' are prescriptions for practice, which I refer to as information systems methodologies, that inform systems developers and give rise to concrete thinking and acting.

Many methodologies have been proposed during the last two decades, e.g. Structured Analysis/Structured Design, Jackson Systems Development, ISAC, to name a few.[1] When analysing and designing information systems, practitioners will as part of that have to know about different methodologies and be able to handle them effectively. The topic is presented below from four different view-points.

Non-Universal Methodologies

Many organisations have chosen to commit themselves to one methodology as an organisational standard to be followed by all development

[1] (DeMarco 1978; Jackson 1983; Lundeberg *et al.* 1979), respectively.

projects. The British Civil Service use SSADM, many Danish banks use SA/SD, etc.[2]

The NNA-Case illustrates what can happen when a single methodology is taken as standard.[3] NNA is an organisation that develops information systems for the Danish savings banks. NNA has taken SA/SD as a standard to be used in all projects. The savings banks are separate companies and therefore in competition. Projects describe the information systems by means of logical data-flow models without making physical data-flow models. As a consequence:

- There is little understanding of the banking practice, because the logical models (and the computer systems) are common to all the saving banks, though their banking procedures are different.
- There is a focus on computer systems rather than on the information systems.
- The properties of the information systems change during the whole process of development and many mistakes are made.[4]

These consequences are very much related to the general conditions for information systems analysis and design in NNA. The information systems are common to all the savings banks and, at the same time, there is competition amongst the savings banks. Because the information systems are based on common ideas much local variation in banking practice is not considered relevant. Furthermore, it is difficult to reveal the variation to NNA because of the competition. The cooperation between NNA and the savings banks is formal and bureaucratic where the data-flow models are produced by NNA and used by savings bank staff for decision-making about the future information systems rather than utilising the models as a means in a process of mutual learning between NNA and the savings banks.[5] Even if SA/SD is an inappropriate methodology in most development efforts in NNA, the methodology has been chosen as a company standard with considerable effort and expense in educating the systems developers in building data-flow models.

The first problem area is based on the following assumption.

Assumption 1 *Information systems methodologies are not universal.*

[2](DeMarco 1978; Yourdon 1982) and (SSADM 1984).

[3]The NNA-Case stems from (Andersen *et al.* 1984). The NNA-Case is used as a small illustration and is not part of the research reported in this thesis.

[4](Andersen *et al.* 1984, p. 22).

[5](Andersen *et al.* 1984, p. 23).

Most methodologies contradict this assumption implicitly as they never question whether they are applicable and appropriate in all situations in information systems analysis and design. A universal methodology would, of course, be a grand solution to many problems (including those in NNA), but the idea is not feasible. Methodologies are not universal. They relate to different problems which are all relevant in information systems analysis and design, some to organisational change, some to design of computer systems, and yet others to programming and database implementation. The idea of making a super-methodology by putting together the best of each methodology is based on several unrealistic assumptions. Firstly, it is assumed that we can assess which is the 'best' methodology in all parts of information systems analysis and design. But in some situations it would be inappropriate to use data-flow modelling, for example, because it is a situation characterised by issue-based work which is difficult to capture in a data-flow model while in other situations it would be a very good idea. Secondly, it is assumed that the 'best' bits of the methodologies can be put together and form a meaningful cohesive set of guidelines. Suppose we have found that DeMarco's data-flow modelling is the best methodology for analysis of the existing organisation and that Jackson's entity modelling is the best for designing an information system to that organisation. However, data-flow modelling and entity modelling are based on very different perspectives on the real world and they cannot be integrated without radically changing one or the other.

We will have to manage without a universal methodology. This leads to one of the research issues of this thesis.

Issue 1 *How can the analysis and design of information systems be done by means of non-universal methodologies?*

Difference among Methodologies

The second problem area is based on the following assumption.

Assumption 2 *A choice among information systems methodologies is always made but in order to make a feasible choice it is necessary to know about significant differences between methodologies.*

NNA chose to use DeMarco's data-flow models, other organisations choose other methodologies. Each project faces a choice of methodologies for their particular situation,and some projects may even have to apply more than one methodology because no single methodology covers all needs in the project. No matter how we look at the use of methodologies a choice

is always made. The choice of methodologies may be made explicitly or implicitly, nevertheless, the actual use of a methodology represents a choice.

It is not uncommon that systems developers seek to use a single methodology they know well and have tried before. In a situation of choice there is a tendency to use the well-known methodology, even though in that particular situation another methodology may well be more suited. The observation underpinning this is that the repertoire of many systems developers is limited (often only containing one methodology) and for many reasons methodologies within the repertoire are preferred to entering the uncertainty of using a methodology yet to be learned.

The different methodologies available form a varied and motley spectrum. It is necessary to make some sense of this mess not only in research but also in development projects and organisations. It would be valuable to be able to compare methodologies or at least have a conceptualisation of the differences. The practical need to understand these differences forms a research issue.

Issue 2 *What are the relevant difference between analysis and design methodologies and how can these difference be conceptualised?*

Different Methodologies for Different Situations

It is not only methodologies which are different. Situations are also different and they call for different methodologies.

Assumption 3 *Different information systems methodologies are useful in different situations.*

It was not commonly known in NNA that SA/SD was inappropriate as a standard in their organisation. NNA did, of course, compare SA/SD with other methodologies before selecting it as a standard, but it is the means of finding out whether SA/SD is suited and appropriate for all situations of information systems analysis and design that are inadequate.

Issue 3 *What are the relevant relationships between different information systems methodologies and different situations?*

It is necessary to find out when it is appropriate to use a specific methodology and, especially, when it is not. The usefulness of a methodology denotes the aspects of information systems analysis and design where the methodology provides substantial (not necessarily detailed) advice and support to the systems developers.

Understanding the consequences of adapting a particular methodology to a specific situation, i.e. knowing about the usefulness and the limitations of specific methodologies, is a sign of deeper insight than a simple understanding of differences between methodologies.

Multi-Perspective Approach

Assumption 4 *Combination of information systems methodologies to multi-perspective approaches is a sound ideal.*

By applying a variety of methodologies featuring different angles, views, aspects, and perspectives it may be possible to facilitate a more thorough inquiry in analysis and design of information systems. In theories about analysis and design this is called multi-perspectivation.[6] It is put forward as an ideal that inquiry ought to be based on several perspectives. Insight gained through the use of one methodology causes changes and alterations of understanding based on other methodologies.

The idea of multi-perspective approaches is close to common sense and intuition, but it becomes increasingly problematic and complex when we want to find out how it can be practised, i.e. above the level of 'look at the other side of the coin', 'get a second opinion', and 'play the devil's advocate'. What happens if we see a methodology as a promoter and defender of a view-point and at the same time as a challenge to other view-points? Some methodologies may not be a challenge to each other, while some combinations of methodologies may create debate and even combat between ideas.

Issue 4 *How can the use of multi-perspective combinations of information systems methodologies be practised?*

The combination of methodologies cannot be made at random and certainly not uncritically. Some combinations of methodologies may be problematic as, for example, if there is not really an interplay and a challenge between the methodologies. What is then a useful combination? Are there problematic combinations? How can we manage the methodologies and their interplay?

1.2 Research Approach

Action research and soft systems ideas play a major role in the research reported. By presenting the research tradition in Lancaster the applied research approach can be understood.

[6] Cf. (Nygaard and Sørgaard 1987).

Action Research and Soft Systems

The origin of the research approach taken to deal with the problem areas outlined in the previous section is the research tradition of the Department of Systems and Information Management at Lancaster University. The Lancaster tradition is designated by its commitment to action research and to soft systems ideas. I take the same standpoint.

Standpoint 1 *Action research is taken as the prime idea behind the research approach.*

Action research denotes a whole range of different approaches, some more action than research and others more research than action. It has generally been defined as research which aims to contribute both to handling practical situations and to the goals of research by means of collaboration within a mutually accepted framework.[7] The Lancaster research tradition is action research in this sense.

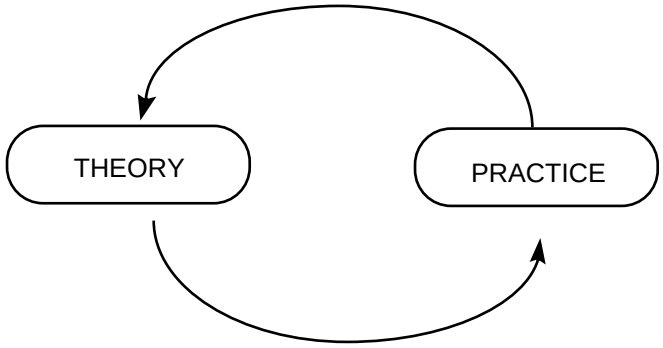


Figure 1.1: *The learning cycle of action research.*

Figure 1.1 describes the never-ending learning cycle of action research as it is seen in Lancaster. Each year a number of projects are undertaken where systems ideas are validated by applying them in practice. Reflections on practice lead to systems lessons that become part of or change the soft systems theory.[8]

Even though the theory and practice of action research are equally important the usefulness of the lessons can only be evaluated by ensuring that they are elicited from practice and by applying them in practice. Applying the systems lessons, i.e. the methodologies, in particular situations serve as the ultimate test. Checkland writes:

[7](Rapoport 1970; Warmington 1980).

[8]See for example (Wilson 1984, p. 2), (Checkland 1981, p. 151).

> its criterion of success was that the people concerned felt that the problem had been 'solved' or that the problem situation had been 'improved' or that insight had been gained.[9]

Soft systems ideas, concepts, and methodologies are the outcome of the research in Lancaster.

One of the problems with action research is that it cannot be fully planned and the outcome cannot be anticipated. The specific setting conditions what can happen in a project if we are to do something useful in the given situation. Another problem is that the lessons that can be learned from action research are hard to generalise. In an experiment in a laboratory generalisations are made on the basis of general criteria, e.g. repeatability, but in action research generalisations can only be made on criteria specific to the lessons and the situations where the lessons arose. That is, generalisations can only be made when they are supported by specific *argumentation*.

The research in Lancaster has been on soft systems ideas. In my research I have taken these ideas as given rather than questioning them.

Standpoint 2 *Soft systems ideas are taken as a theoretical foundation for the research.*

Soft systems ideas and Soft Systems Methodology[10] (SSM) in particular are based on the notion of human activity system. Human activity systems are assumed *not* to exist in the real world. Instead, we may inquire into the problems of the real world by formulating views on the real world in terms of systems, i.e. wholes, and then by comparing these systems views with the real world we learn about the real world.

The ideas of SSM have been applied in various ways throughout the research.

Research on Information Systems Methodologies

When I began the research I was convinced that what was needed was a classification of methodologies for analysis and design. More specifically: How can methodologies for analysis and design be classified, such that the classification, in a detailed and distinct way, reflects important aspects of the potential application of the methodologies?

In this vein of thinking it became important to find a set of concepts and criteria by which methodologies could be distinguished. On the one

[9] (Checkland 1981, p. 146)
[10] (Checkland 1981).

hand, the concepts should be sufficiently general to apply to all methodologies. On the other hand, they should also be sufficiently specific to signify each of the methodologies. A balance between the general and specific would have to be found. A feasible way of finding that balance seemed to be by means of action research. Only by eliciting the classification from practice and continually confronting the classification with practice could a proper balance be found. A proper balance in the classification was thought of in terms of how useful it would be for a practitioner. Systems thinking seemed to be a way of encapsulating the concepts and the criteria to something coherent, i.e. a classification.

The intention was to come up with a methodological framework for comparing methodologies. Then I would apply selected typical methodologies in suitable situations and finally I would evaluate the usefulness leading to a classification of the methodologies. In the spirit of action research a considerable iteration between the practice of selecting and applying the methodologies and the theory of re-evaluating, re-classifying, and re-making a framework was going to take place.

The initial approach had disadvantages. Firstly, the approach was not feasible within the time-scale of a Ph.D. study. Secondly, as it turned out, the research showed that the basic need was not for yet another classification of methodologies.

Thus re-thinking the whole research and taking broader research issues as outlined in Section 1.1 lead me to rely on three different but related cases.

Case 1: This case is based on interviews with a few systems developers about their views on information systems methodologies. The case gives insight into methodologies at the individual level. The insight is of the type: how systems developers view methodologies, personal and professional values and preferences, etc.

Case 2: This case is based on the use of three different methodologies in similar situations. From this we can learn about methodologies at the project level. This case is the one closest to the initial intentions and the lessons that can be learned from this approach are of the kind: data-flow descriptions are insufficient when developing computer-based support for issue-based work while Newman's office models are useful, and these models are useful together with Jackson's methodology, etc.

Case 3: This case is about methodological learning at the organisational level. The research was organised as an intervention into a

DP department with the aim of investigating how new methodologies and methodological knowledge could be introduced to and learned by the systems developers in the department.

The three cases are not only different with respect to the level of methodological practice they relate to. They are also different with respect to: the amount of resource used, the lessons that have been learned, the generalisations which can be based on them, and the nature of the conclusions.

Case 1 was produced with relatively little effort as I interviewed eleven systems developers from the same DP department about their views on the use methodologies in information systems analysis and design. The interviews produced examples of statements about methodologies more than a representative picture of how reality really is. There is a narrow limit as to how far these statements can be generalised. Three lessons have been learned from these interviews.

Case 2 is based on the work of six colleagues and myself. Each of us took part in one of three projects that ran simultaneously. Each project were given similar conditions but different methodologies and three months of effort to carry out an analysis and design project in a bank. Sixteen lessons have been learned from this case. Some of these lessons concern the specific methodologies that were used. Other lessons are about the different ways of using methodologies in projects.

Case 3 is based on an intervention of more than six months. During this period I intervened into the practice and the problems of a group working with the introduction and learning of methodologies in a DP department (the same as in Case 1). The lessons that have been learned from this are of a much more fundamental nature than lessons from the two previous cases. Six lessons concerning introducing, learning, and choosing methodologies came out of this effort.

In summary, the study reported in this thesis is characterised by a specific set of assumptions about the topic and a specific choice of approach. The results of the research are very much a consequence of these choices.

2

Methodologies in Information Systems Analysis and Design

The previous chapter outlined the subject of the research reported in this thesis. This chapter establishes and expresses an appreciation of methodologies and information systems analysis and design. This appreciation serves as a frame of reference throughout the thesis.

Section 2.1 presents briefly seven methodologies in order to show some of their obvious features and differences by looking at them simply as phenomena. A conceptual foundation of information systems development is given in Section 2.2. Section 2.3 offers a discussion and definition of the concept of methodology in information systems analysis and design.

2.1 Information Systems Methodologies

The seven methodologies to be presented in this section display basically different assumptions, values, and features. The presentation is done in brief and with specific reference to the unique characteristics of each methodology. I have chosen three categories to express these characteristics, namely: description of technical systems, organisational analysis, and participation. Within each category I have selected a few very dif-

ferent methodologies to show some of the variety rather than common features.

Description of Technical Systems

The methodologies associated with technical systems have prevailed over the longest period, but have also received the strongest criticism in the field of information systems analysis and design.

Data-Flow Models

Structured Analysis/Structured Design (SA/SD) is probably the most well-known of the methodologies for description of technical systems. It was developed in the late seventies.[1]

SA/SD is often referred to as data-flow description because the fundamental concepts are data, process, and flow. The basic technique is description by means of these concepts in data-flow diagrams. A data-flow diagram is a graphical representation comprising four elements: data-flow (→), process (◯), data store (=), and source/sink (□). A data-flow diagram is strictly hierarchical. At the top-level only one process is described with its input and output data-flows. The top-level process is then recursively detailed further into more specific processes, flows and stores. The decomposition ends when all the low-level processes are sufficiently detailed and can be described in Structured English. Each level in the hierarchical diagram is consistent with the level above. The philosophy is that the top-level and the low-level describe the same phenomenon but at different levels of detail.

Data-flow diagrams are used to provide simple structure to a complex phenomenon by the principle of top-down decomposition. Firstly, a diagram is made of the physical data-flow in the existing organisation depicting the concrete level including people and the documents they handle. Secondly, the diagram of the physical level is transformed to a logical level describing what the system is doing rather than how. The second diagram is consistent with the first diagram thus describing the existing organisation in terms of data-flow. Thirdly, the logical diagram is transformed into a new logical diagram that describes a new data processing system. The new system is improved according to the users' desires for change. Fourthly, the logical diagram of the new system is

[1] The basic concepts in systems specification by DeMarco (1978), the design parts by Yourdon & Constantine (1979), and Yourdon (1982) made it into an all-embracing methodology.

transformed back to a physical equivalent as it is decided which processes in the system should be performed by a computer.

In the design activity the new physical diagram is used as a basis for dividing the computer system into manageable components (modules). The interaction between these components in terms of inputs and outputs are described in Structure Charts.

Entity-Relationship Models

Database methodologies are somewhat different from each other. I have chosen to present a methodology that I think illustrates the basic ideas of database design nicely. Information Modelling (IM) is inspired by entity-relationship modelling, but in addition it provides a set of more precise definitions of the central concepts.[2]

The purpose of IM is to systematise and logically organise data and collections of data in a database. The structure of the database is derived from an information model. The core of an information model is the entity-relationship diagram where entities (□), sometimes called objects, denote anything that can in principle be named, such as person, place, thing, event, or concept, and relationships (□–◇–□) denote an association between entities. In drawing the diagram it is important to seek 'well-defined' entities and relationships in order to ease the step from the information model to the database. It is possible to place entities in a hierarchy by means of the standard relationship: super-type, e.g. the entity 'vehicle' is a super-type of 'air vehicle', that in turn may be a super-type of 'helicopter'. Other standard relationships, such as associative type, sub-type, and characteristic type are used in the same manner to describe other modes of abstraction.

By detailing the descriptions of entities and relationships a fully defined information model is obtained and a structure of the database can be derived from this. Detailing can be done either by defining other and more detailed entities and relationships or by defining the data necessary for the entities and relationships. The necessary data can be: purpose, properties, effect, etc. Having a well-defined information model this is close to a definition of a data base. In a relational database, for example, a typical implementation is made by making a table for each entity and each relationship in the model.

[2] Chen (1976) came up with the original entity-relation model. IM was developed by Flavin (1981).

Dynamic Data Models

Jackson System Development (JSD) is a methodology for developing flexible and maintainable software systems.[3] It is based on a distinction between a model of reality and systems functions. Jackson writes:

> It is a fundamental principle in JSD that the developer must begin by modelling ... reality, and only then go on to consider in full detail the functions which the system is to perform. The system itself is regarded as a kind of simulation of the real world; as the real world goes about its business, the system goes about simulating that business, replicating within itself what is happening in the real world outside. The functions of the system are built upon this simulation;[4]

The model of reality, called the Initial Model, is built as a set of communicating processes. Each real world entity of interest to the system is modelled as a sequence of atomic actions. Entities ($\Box$) in the model are coupled to reality and to each other by two types of data communication: data stream connection ($\Box \rightarrow \bigcirc \rightarrow \Box$) and state vector connection ($\Box \rightarrow \Diamond \rightarrow \Box$). In a data stream connection one entity is sending a stream of buffered messages (data) to another entity for the other entity to read when appropriate. In a state vector connection one entity takes the initiative and reads the state of another entity without influencing the other entity.

Once the Initial Model has been built, functions are added to the model. The principle is that the functions use the data in the model without altering its state. Functions are described in the same way as entities (and are thought of as entities) and they relate to the model by the same two types of connections. If, for example, a model describes a bank by the entities 'customer' and 'account' they denote the stable and relevant aspects of the reality—the aspects that are not likely to change. A function based on such a model could be the production of a monthly report of the balance of all accounts. Jackson claims that functions are more likely to change than the reality where they are used and should therefore be easy to change without changing the underlying model.

JSD covers information systems development all the way from design and specification phases to implementation phases where the model and the functions are implemented on a computer. The descriptions made in the first phases are to a large extent specifying the behaviour of the system. During the implementation phase the specification is translated into a set of executable programs.

[3] (Jackson 1983).

[4] (Jackson 1983, p. 4).

Organisational Analysis

Methodologies for organisational analysis in information systems development can alleviate some of the problems imposed by more traditional methodologies for description of technical systems. Avison and Fitzgerald have identified some of the consequences of applying traditional methodologies: instability and inflexibility of the computer systems, user dissatisfaction, unambitious systems design, etc.[5] Some of the methodologies for organisational/informational analysis seek to overcome some of these shortcomings in different ways and for different reasons.

Information Systems Analysis and Change

Information Systems Work and Analysis of Change (ISAC) covers most aspects of information systems development.[6] It does so from an organisational perspective. Information systems are seen as organised cooperation between human beings with the aim of processing and conveying information to each other. The three main activities of ISAC are:

- analysis of the organisation's need for change, information and computers,
- information analysis, and
- computer systems design and implementation.

During the first analysis, the organisation's problems and the interest groups of the organisation are identified, investigated and relationships between problems and interest groups are found. ISAC acknowledge that problems relate to interests in the sense that they are seen as problems from different view-points. In order to understand the organisation's need for change the existing organisation must be understood in terms of visions held by interests groups and the objectives of interest groups. A-graphs are made as a somewhat more 'objective' foundation for the analysis than the messy network of problems and interests. A-graphs are used to describe the activities that relate to problems and interest groups. An A-graph is a semi-formal description of the inputs and outputs of an activity in terms of material and information; it describes the transformation of inputs into outputs in terms of flow and activities. The final outcome of this first analysis is a decision on what to change and how to affect the change.

[5] (Avison and Fitzgerald 1988, p. 21–26).
[6] (Lundeberg *et al.* 1979).

If the identified need for change suggests a change of the information systems, the A-graphs are brought to a fairly detailed level by describing all the sub-activities in yet other A-graphs. When such a level of detail has been reached which allows for the separation of information flows and processing from the rest of the flows and activities, the information subsystems are identified. For each of these subsystems an appropriate level of ambition is found; and the formalisable subsystems are described in I-graphs and C-graphs showing the formal details of information sets and their relationship (in information and component graphs). This concludes the information analysis of ISAC.

After having analysed the information subsystems the computer systems that do the data processing needed are designed and implemented. This involves data structure design, file and database design, design of the overall structure described in D-graphs, design of manual routines, and adaptation of design to particular equipment.

Soft Systems

Information Systems Methodology (ISM) deals with information systems analysis in a systemic way, i.e. from a soft systems perspective.[7] In ISM information is viewed as inseparable from its use and therefore it can only be understood and analysed by looking carefully at the organisation where its use is considered to be meaningful. An existing information system is analysed by making several systems models of the organisation which eventually become amalgamated into one model after some debate. This model is used to examine how information is provided and supports the activities of the organisation. A device, called the Maltese Cross, is used to map the input and output of each activity in the model onto the information processing procedures (some of which are computer systems). In the Maltese Cross it is possible to analyse and identify misfits and mistakes in the relationship between the activities in the organisation and the information processing procedures, e.g. that the same information is processed by two different procedures, that information processing needs to be improved, or that some information is not processed at all.

The purpose is to inquire into information in the organisation by explicating and illuminating different views of the organisation and thereby on information provision. On the other hand, not much attention is given to the development of the computer systems and thus the rela-

[7] (Wilson 1984).

tionship between the technical aspects and the organisational aspects of the computer-based information system is not considered in any depth.

ISM is rooted in and can be seen as an application of SSM in information systems analysis and design.

Participation

Participation as a main issue has been a primary source of inspiration in Scandinavian information systems research for almost two decades. In the U.K. the most prominent representative of the participatory tradition is ETHICS. Recent years have shown that prototyping is a promising approach to participation.

Socio-Technical Approach

Effective Technical and Human Implementation of Computer-based Systems (ETHICS) is a methodology inspired by the Tavistock experiences with 'leaderless' groups, democracy and systems thinking. The experiences are varied and cover a project in coal mining and trade union projects in Norway. This has led to the current approach for socio-technical development of computers and work.[8]

ETHICS strives at two things: job satisfaction for users, and participation.[9] Job satisfaction is the ultimate goal and participation is the most important means to achieve this. The development of computer-based systems is a change of work and organisation. It introduces and surfaces conflicts of interest between the actors in the situation.

> The successful implementation of new systems is therefore a process of negotiation between the affected and interested parties.[10]

Design groups are formed to undertake either representative participation or consensus participation. Consensus participation is the strongest form of participation. The task of the design group is to design: the user-interface, what software and hardware to use, work-place organisation, and new responsibilities. Initially the design group may not be able to do this, but they are trained and educated in order to be able to influence the work in the groups. The role of systems developer is that of a neutral facilitator of the process.

[8] (Mumford 1987).
[9] According to (Avison and Fitzgerald 1988, p. 230–235).
[10] (Avison and Fitzgerald 1988, p. 233).

ETHICS involves a number of stages. Land *et al.* present it as four procedures:[11]

- Two aspects of the social system are described: (a) the essential organisational system in terms of problems, boundaries, objectives, operations, and variance; (b) the essential human system in terms of roles and job satisfaction needs.
- Discrepancy analysis establishing the need for change.
- Future analysis.
- Objective setting and strategy evaluation.

Prototyping

Prototypes are used for explanatory and experimental purposes during system design and realisation. This enables a direct feed-back from the users' insight into the daily use of the system to the developers. The qualities of prototyping also encompass the belief that users can only gain such insight by actually trying out the system in question and not simply by thinking and talking about it.[12]

Software Technology for Evolutionary Participative System Development (STEPS) is a methodology based in the prototyping tradition.[13]

STEPS as a methodology is based on the finding that:

> The competence needed for designing technology and work, and in particular for determining user-oriented quality criteria, is not possessed unconditionally either by the users or by the developers. Their joint participation in the process constitutes a necessary condition for creating the new knowledge required and for a shared experience for design.[14]

STEPS designates information systems development as a cyclic process where information systems evolve by developing a version of a system followed by the use of it and in turn leading to a revision and a new version of the system. There are four activities in the production of a version. Firstly, a version of the system is designed by users and developers in collaboration. This involves the creation and evaluation

[11] Different stages in different references; see (Olerup 1989). The four procedures are from (Land *et al.* 1980).

[12] Cf. (Budde *et al.* 1984).

[13] (Floyd *et al.* 1989).

[14] (Floyd *et al.* 1989, p. 9).

of visions about the quality and the structure of the version and about the associated work. Secondly, a specification of the designed version is made by the developers. Thirdly, the system is realised on a computer by detailed specification, implementation and tests. Fourthly, the embeding of the version into the sphere of work and organisational operations is prepared by the users.

After the production phase, the system version is used by the users in their daily work and maintained by the developers. Finally, this will lead to a need for a revision. A revision is then established with due regard to the overall concept of the system and the project strategy.

2.2 Information Systems Development

This thesis is about methodologies in information systems *analysis and design*. Analysis and design are the 'reflective' activities of information systems development, i.e. the activities that embrace the thinking about the information systems and about their change rather than the actual change of information systems.[15] A conception of information systems development is provided as a context for understanding the methodologies.

Information Systems Development as Human Activity

Information systems development can be seen as a human activity system. SSM provides a framework and a number of techniques for expressing such a view-point as a system and as a set of necessary activities.[16] Figure 2.1 shows information systems development as a system.

Root Definition of Information Systems Development An organisation-owned system for professional systems developers and organisational partakers (users in particular) to develop information systems by intervention into the organisational situation with the purpose of improving information processing including the use of computers.

The five activities in the conceptual model are:

[15] (Andersen *et al.* 1990). Andersen *et al.* take information systems development to consist of the following activities: analysis and design (reflective), realisation (change), and management and communication activities.

[16] (Checkland 1981; Wilson 1984).

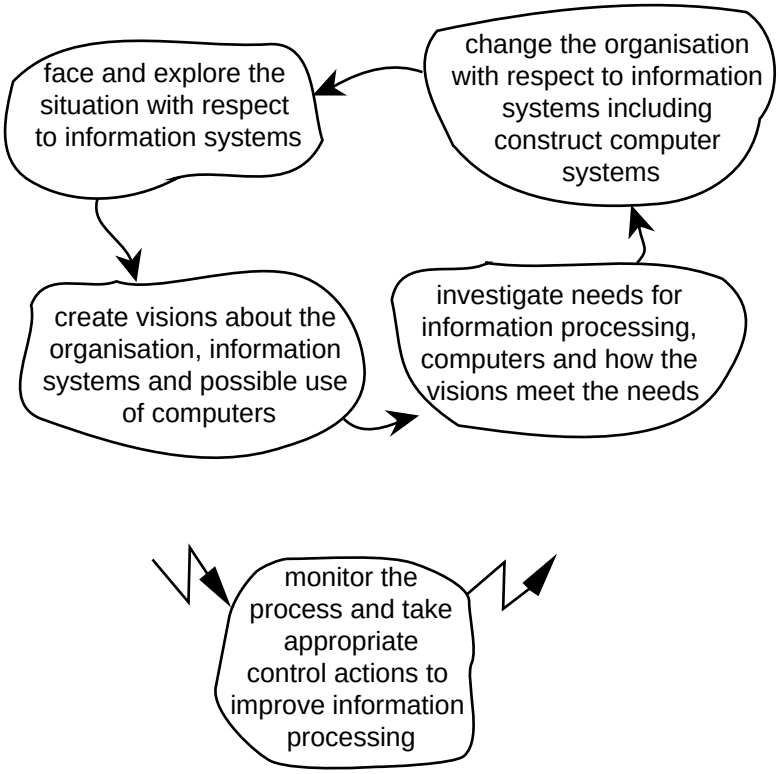


Figure 2.1: *A system of information systems development. The conceptual model is defensible against the root definition. Key: ◯: activity; →: dependency between two activities; ⇝: dependency between an activity and all other activities.*

Face and explore the situation with respect to information systems: In order to find out what may need to be changed in the situation at hand it is necessary to identify areas where improvement is possible. This activity takes the totality of the situation and the outcome is an understanding of which direction the development will go along. By facing the specifics of the situation in terms of organsation, roles, information and computer support insight is gained. This activity is tightly related to the specifics of a particular situation and is carried out in processes like: interviewing users and other partakers, describing initial impressions of discrepancies between the present situation and a desirable situation, and getting acquainted with the situation and in particular the users and other partakers.

Create visions about the organisation, information systems and possible use of computers: Possible future situations are envisaged in order to deal with the problems in the situation and improve the situation. This activity is carried out in processes like: interviewing users about desirable: work and information processing hereto, formulating and modelling a future organisation, design of information systems and computer support, describing computer systems including functionality, databases, interfaces.

Investigate needs for information processing, computers and how the visions meet the needs: The limitations and the flaws of both existing information systems and visions about new are investigated. This activity encompasses processes like: inquiring into and modelling: the constraints of the situation, the organisation, necessary information, and available computer equipment. The typical processes in which this activity is done is by making models of the existing organisation or information systems according to a conceptual framework.

Change the organisation with respect to information systems including construct computer systems: Having found out what can and must be changed at the level of ideas the organisation will actually have to be changed accordingly. While the previous activity basically is an inquiry and a learning process this activity is more acting than thinking — acting with the purpose of bringing about the change envisioned. This activity is carried out in processes like: programming, structuring the database, introducing new information systems, training and educating the users.

Monitor the process and take appropriate control actions to improve information processing: The performance of the other four activities are monitored to gain an understanding of what has been done and of the distance to the desirable performance (typically expressed in a plan). Based on this evaluation appropriate control actions are taken, if necessary, reinforcing or changing the main ideas and the plans of a particular process of information systems development. Appropriate reinforcement is done by changing the performance of the other four activities to the extent this is possible, and to the extent this is not possible the plans or even the main ideas must be changed.

Three Metaphors in Information Systems Development

The above human activity system is fairly general. Actual information systems development processes will map onto this model in different ways. Different theories, ideas and methodologies will also map onto this model in different ways. SA/SD, for example, faces and explores the situation by a survey activity where the project remit is established. Visions are created by making one data-flow model of the new organisation. The investigation is made by making data-flow models of the existing organisation in terms of data-processing. The organisation is changed by implementing this model directly.

Three metaphors have been chosen to understand the variety of relevant perspectives on information systems development: technical development, organisational change, and inquiry process.

Seeing information systems development as technical development is to focus on the complexity of the computer aspects related to information systems. The technical challenge involves: making computers work properly, structuring databases efficiently, defining functionality precisely, and designing an effective interface. Computers have the advantage and disadvantage of requiring precise and formalised descriptions, namely programmes. The skill and insight it takes to formalise matters that are not from the outset formal, is not negligible. In this view it is important to be able to master the computer aspects of the activities in the above systems model, e.g. the construction of the computer system is the most important aspect of the 'change'-activity. As a consequence, the activities concerning 'face and explore' and 'control' get only little recognition.

The theories and ideas within this view are typically based on conceptions of what a computer is and how it can be programmed. Mostly, they are formulated as methodologies.[17]

Seeing information systems development as organisational change is to take a starting point in the complex organisational setting. Information systems contribute to organisations by processing, storing and retrieving information which is necessary to do the tasks of the organisation. It is assumed that if the organisation can be understood then it is possible to arrive at an improved information system for the organisation. This may be done by identifying organisational roles, activities, stakeholders together with the associated information 'needs'. The tech-

[17]The methodologies for description of technical systems are examples of this. Software Engineering is the discipline where the technical view is applied most frequently, cf. (Pressman 1977).

nology, i.e. the use of computers, is considered important in this view, but it will always remain a means for achieving some organisational goals.

The organisational view is to a large extent based on theories and ideas about organisations and from these ideas about the use of information in organisations are derived.[18]

Seeing information systems development as inquiry is to focus on processes of understanding and learning. An inquiry is taking place in order to provide insight into the existing organisation, the technical possibilities, and in order to create ideas about desirable changes. In this view, information systems development is not only seen as technical and organisational but also as processes of humans that think, discuss, learn, communicate, negotiate, debate, etc.

The inquiry-view is typically based on the application of general philosophy or epistemology of information systems development. The theories within this view advocate important concepts, their use and their relationship.[19]

In this thesis, all three metaphors will be used from time to time. However, irrespective of the metaphor used, information systems development is taken to include the following four elements:

- The development is situated in a *user organisation*. The information systems are tailored to the particular needs of the organisation and of the users. The information systems are supporting work (or parts thereof) carried out in that particular organisation.

- It is the development of *information* systems rather than computer systems.

- There is, however, a focus on *computers* in developing information systems, i.e. on computer-based information systems.

- The development is primarily done by professionals, that is, *systems developers*. They will typically be organised in a project with the overall responsibility for the development and for involving other partakers.

[18] E.g. Ciborra (1981) has taken an organisation theory, (Williamson 1979; Ouchi 1980), and applied it to information technology. Other examples are: (Mathiassen 1981; Kling 1987).

[19] Cf. (Mathiassen 1981) applies dialectics (Israel-version), (Stage 1989) applies dialectics (Bohr/Israel-version), (Ehn 1988) applies Heidegger, (Lyytinen 1986) applies Habermas, to name a few.

It is worth noticing that the term 'information systems development' is apparently used in a hard way. Ideally, in the soft systems tradition, it should be called 'the development of processes and structures that we may look upon as an information system'. This seems unnecessarily tedious and causes a break-down of the language. Therefore, 'information systems' is used in its soft sense though it may sound hard to some. Similarly, a 'systems developer' is merely a professional taking part in information systems development.

Notice also that this section has been about information systems development rather than 'information systems analysis and design' which this thesis is about. This has been done to provide a comprehensive view of the working practice referred to throughout this thesis. To focus the thesis on analysis and design is based on the logical distinction between the reflection aspects and the change aspects of the above systems model. By choice, this thesis is about methodologies used in analysis and design where analysis and design is taken to be the reflective aspects of information systems development.

2.3 The Concept of Information Systems Methodology

The concept of information systems methodology is fundamental in this thesis. Until now this chapter has discussed what a methodology is by presenting examples and by conceptualising information systems development. Now, the discussion is brought to a conclusion by defining and characterising what is to be taken to be 'a methodology' in this thesis.

Conceptual Definition

Many and very different definitions of the concept of methodology can be found in the literature. Some tend to find that methodology is a sign of malady. Others see methodology as the only way of transcending a believed crisis in information systems analysis and design.

Lyytinen & Lethinen call the belief in and use of methodologies, in general, a mortal sin.[20] They argue that to rely on methodologies is founded in a mechanistic world-view where information systems development is seen as a set of deterministic model transformations guided by a methodology which guarantees against human errors and influence.

[20] (Lyytinen and Lehtinen 1987).

From this view-point, a methodology is an algorithm. Using such a restricted and narrow definition of 'methodology' makes it easy to argue that the vision of methodologies purports information systems development as a fully rational process, which it is not.

Another belief is expressed in the introduction to the manual of the methodology SSADM. Here it is stated that a methodology (SSADM in particular) is:

> Like a good kit of tools, it provides the right tools to enable each development task to be successfully completed.[21]

It is probably such statements that Lyytinen & Lethinen are worried about. The pursuit, as I see it, must be to find a concept of methodology that is between the rejection of and the blind faith in methodologies. Furthermore, we must find a concept of methodology that is useful for expressing relevant aspects of methodologies and their relationship with information systems analysis and design.

Mathiassen sees methodologies as the part of our knowledge about information systems development that provides guidelines for thinking and acting.[22] The guidelines of a methodology fall into three categories:

- *Techniques* are proposed lines of action. A technique relates the outcome of processes with knowledge about how the processes can be carried out, e.g. programming by step-wise refinement. Often a technique is detached from considerations about necessary resources and relevance.

- *Tools* are artefacts used as means by a technique to achieve the intended goal efficiently.

- *Principles of organisation* are relations between the process of development and its environment, what resources are needed and how they can be utilised. The principles are concerned with the cooperation of many and different actors.

Mathiassen's concept of methodology represents an improvement compared to that of Lyytinen & Lethinen and of SSADM. On the one hand, methodologies guide the systems developers in developing information systems. On the other hand, they are based on knowledge about the process of development.

[21] (SSADM 1984).

[22] (Mathiassen 1981, p. 98–101). Actually, Mathiassen (and with him many others) use the term 'method' to denote what in this thesis is termed 'methodology'.

Checkland sees methodologies differently.[23] To Checkland SSM is intermediate in status between a philosophy and a technique. A philosophy is a set of non-specific guidelines, that is, a philosophy will address the question of *what* to do and sometimes it will even address the question of *why* to do it. A technique is a precise and specific programme of action which will guarantee a particular and anticipated result, e.g. proving properties about a small computer program by Hoare-logic. This notion of technique is similar to Mathiassen's, i.e., *how* can the processes be carried out. A methodology lying between philosophy and technique contains elements of both *what* and *how*.[24] Furthermore, Checkland seeks to characterise SSM by the following:

> it should be capable of being *used* in actual problem situations; it should be *not vague* in the sense that it should provide a greater spur to action than a general everyday philosophy; it should be *not precise*, like a technique, but should allow insight which precision might exclude; it should be such that any developments in 'systems science' could be included ... and used if appropriate in a particular situation.[25]

Information systems methodologies as, for example, those presented in Section 2.1 reveal that some are close to technique and a few are close to philosophy, but they all rely on a balance between the two. The latter criterion, though, can only be met by a few methodologies and I shall refrain from applying it in general. So, a methodology must be usable and simultaneously be not vague and not precise.

Methodologies are not the same as practice. This is an important and sometimes neglected point. The distinction between methodologies and practice has been made by Mathiassen and later in the MARS-project.[26] Working practice is what is actually done in a situation in information systems development. It is the concrete actions taken while a methodology is an abstraction, i.e. an intellectual construct. A methodology may be a generalisation of the working practice in one or hopefully several situations. As a generalised construct it may inform practitioners about a possible change of working practice. The desire to apply a specific methodology in a particular situation may influence the working practice which in turn may change the situation. The situation may give rise to a

[23](Checkland 1981). It is mostly about his own methodology, SSM; but it is in many ways a general definition.

[24](Checkland 1981, p. 162).

[25](Checkland 1981, p. 162).

[26](Mathiassen 1981; Andersen *et al.* 1990).

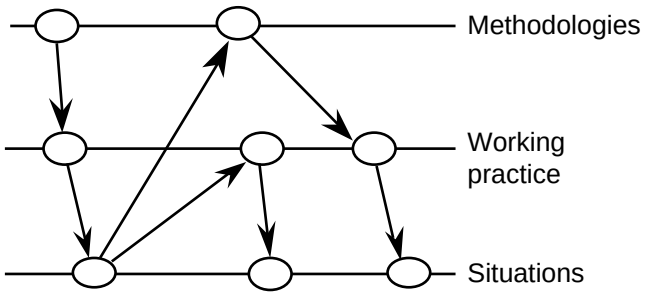


Figure 2.2: *The relation between situation, working practice, and methodologies. Key: ◯: a state; →: influence; time is implicitly moving from left to right.*

change of working practice and the methodologies desired to be applied, see Figure 2.2.

Definition 1 Working practice *is the concrete actions that are actually taken in a particular situation of information systems development. Usually it is the working practice of the systems developers and others that take a professional part in the development efforts.*

We can look at a methodology in terms of Checkland's systems typology as both a designed abstract system and as a human activity system.[27] A methodology is a designed abstract system in the sense that it is constructed to be *used* as an intellectual artefact that needs to be interpreted by systems developers in order to lead to actions. The interpretation is based on the traditions, norms, and values of the interpreter and the outcome has to do with how well the interpreter is able to appreciate and internalise the norms and values of the methodology. Methodologies are designed by designing an overall normative system for action and especially a language and a set of techniques as a means for bringing the normative system into action. The emergent properties of a methodology include the norms and values of the methodology as a whole.

Looking at methodologies as human activity systems means to see a methodology as activity with purposes. The purposeful activity outlined in a methodology is only meaningful in relation to a specific Weltanschauung. The systems outlook is still normative in the sense that it does not purport to describe reality (a human activity system of working

[27] (Checkland 1981, p. 109–122).

practice may). It is a vision of or a prescription for a desirable system. As a methodology contains elements of 'what' it needs to be further detailed by means of more specific activities explaining 'how' before it in principle could be implemented.

The notion of methodology is general. But there is no reason to be more specific than saying that methodologies in information systems analysis and design are methodologies that one way or another are useful in practicing information systems analysis and design. Let us now summarise the discussion on methodology.

Definition 2 *A* methodology *in information systems analysis and design is a general normative system that prescribes whats and hows in relation to actions which are useful in some situation of information systems analysis and design. When a methodology is interpreted and used by systems developers it influences their working practice.*

Notice that one of the implications of this definition is that a methodology is not related to a single situation only. It is a *general* system, i.e. it is thought as being potentially useful in many situations.[28]

Characteristics of a Methodology

A methodology is defined as a whole in Definition 2. In the following I will characterise what I take to be the parts of a methodology.

Definition 3 *A methodology is* characterised *by three aspects:*

- *Domain of use and modelling languages. (Product-oriented).*
- *Frame of action and related techniques. (Process-oriented).*
- *Weltanschauung.*

These three basic characteristics are useful and sufficient for the purpose of this thesis. The characterising aspects are an extension of Mathiassen's characteristic of methodologies.[29]

The *domain of use* of a methodology is the scope or the range of situations within which it is useful. The limitations of a methodology are

[28]The definition is thus different from Atkinson's (1986) where a methodology is taken to be the thinking behind a working practice. Notice also, that methodology is defined without any reference to 'method', e.g. as the *logos* of methods. The definition is a deliberate attempt to avoid the confusion presented in the above discussion.

[29](Mathiassen 1981).

just as important as its usefulness; the domain of use characterises the methodology both with respect to the type of situations and what part of the process of development it is concerned with. Some methodologies may be useful when we are about to develop an overall information strategy for an organisation, some when developing information tools for bank clerks, etc. As already indicated in Chapter 1, methodologies relate to different parts of the process of development, some are useful in analysis, some in design, etc. It is crucial to understand the domain of use of methodologies to avoid relying on a methodology outside its domain.

A *language* of a methodology is a set of concepts and rules for their application. A language can be more or less precise and more or less structured. Most methodologies have a language for making *models* which means that they can be used to explicitly express an understanding of a situation, an organisation, an information system, or a computer-based system.[30] DeMarco uses, in his methodology, the concepts of data-items, data-processes, and data-flows each with their own graphical notation to express an organisation as a data-processing system. The rules for putting these graphical symbols together forms the rules for applying the language.[31] However, most parts of a language remain unstructured. For example, SSM is based on 'rich pictures' containing elements of processes, structures, and climates which are concepts that will always be open to interpretation and discussion, even though the rich pictures are models.[32]

The *frame of action* of a methodology is a set of activities that should or could be carried out when using the methodology. The prescription for these activities are on varying levels of 'whats' and 'hows' as already discussed above. The frame of action includes techniques, i.e. programmes of action that may support one in doing parts of the activities. The frame of action also prescribes a relation between the activities that often will be in terms of a sequence in which the activities must be done or in terms of logical dependencies stating that B cannot be done properly before A together with some advice about iteration. The frame of action relates to the language because it prescribes ways of using the language, e.g. SA/SD offers techniques for inquiring into an organisation in a way that leads to a model expressed in the data-flow language and a few techniques for transforming the model into a model of a new and more efficient organisation.

[30] (Wilson 1984, p. 8, for a definition of model).
[31] (DeMarco 1978).
[32] (Checkland 1981).

The *Weltanschauung* of a methodology is often implicitly given by the other characterising aspects. The Weltanschauung of a methodology is the set of views and beliefs that makes the methodology meaningful. Mathiassen writes on the related concept of 'perspective of methods':

> We choose to let the perspective denote both the coherent understanding of the phenomenon and the understanding we acquire by applying a particular conceptual framework to describe the phenomenon. ... When we choose and apply a particular method then, at the same time, a particular fundamental view on the organisation is chosen.[33]

In this sense, the perspective is inherently connected to a phenomenon. I shall, however, use Weltanschauung in a way where it is related to methodology alone, being aware that it can only be understood by relating it to a phenomenon. The Weltanschauung of a methodology denotes what is focused on using the methodology by being a pattern consisting of norms, values, and concepts that structure our attention and filter our impressions of a situation. Using a methodology, that is using a Weltanschauung, is in this sense an act of exercising this pattern. Thus, using a methodology is more than doing the activities in the frame of action, it is also acting according to or with due respect to the norms, values, and beliefs of the Weltanschauung. This does not mean that we cannot use a methodology without sharing these beliefs. We can, for example, use SA/SD without believing that it is only relevant to examine how data is floated around in an organisation to develop proper information systems. But it is not meaningful to use SA/SD if we do not think that it is useful (to some extent together with other methodologies) and if we do not 'play the game' while we are using it. Being aware of the Weltanschauung of a methodology is to be aware of the implications on values and beliefs while using the methodology.

2.4 Summary

This chapter has in different ways aimed at answering the question: What is a methodology in information systems analysis and design? It started out in Section 2.1 where seven very different information systems methodologies were presented. Each of the methodologies was presented in one of the three categories: technical description, organisational analysis, and participation.

[33](Mathiassen 1981, p. 99, my translation).

In Section 2.2 the understanding of information systems development that is taken in this thesis was given as a human activity system. The contents of this human activity system were presented partly as an elaboration of the activities of the conceptual model and partly as the three metaphors: technical development, organisational change, and inquiry.

The chapter was concluded by a discussion of the concept of methodology. This gave rise to the definition of three concepts: working practice, methodology, and characteristics. The definitions were provided as summaries of conceptual discussions as to be explicit about the central concepts of the thesis.

-

3

Related Research

During the last decade the study of information systems methodologies has taken various forms leading to many and quite different theories, ideas, and approaches to appreciate methodologies. Such studies are at a meta-level relative to methodologies. They have methodologies as the subject area whereas methodologies have information systems as the subject area.

This chapter addresses the meta-level in the sense that it surveys and discusses prior research in the field of appreciating methodologies. The appreciation approaches are ways of going about learning and understanding the features and the usefulness of methodologies used in information systems analysis and design. In the literature they are often denoted as approaches for 'evaluation of methodologies', 'comparison of methodologies', or 'selection of methodologies'.

The survey is conducted as a soft systems inquiry. This means that the survey of the appreciation approaches is done by usingCheckland's SSM.[1] It is based on soft systems ideas, that is, systems are considered *not* to exist in the real world. We may, however, inquire into the problems of the real world by formulating views on the real world in terms of systems, i.e. wholes, and then by comparing these systems views with the real world to learn about the real world.

Only one systems view is used in this survey. By mapping the appreciation approaches I have found in the literature, onto this systems view it is possible to gain insight into these approaches. It is also possible to evaluate the approaches to some extent. Many other views could have been used, but I have chosen a very simple and pragmatic view. Formu-

[1](Checkland 1981).

lating it as a systems view forces me to be explicit about my choice. It is likely that others would have chosen differently. The point is, however, the assertions and the conclusions reached are all defensible against this initial choice.

Section 3.1 surveys the approaches to appreciate information systems methodology. Section 3.2 is a general and theoretical critique of the approaches found in the literature together with some ideas about a possible new approach. The concluding remarks are given in Section 3.3.

3.1 Appreciation Approaches

The fundamental question regarding appreciation approaches, "Which methodologies in which situations?", is chosen as the main focus in the survey. In terms of soft systems thinking this question can be formulated in a root definition:

> *Root Definition Used in the Survey*
> A system for information systems researchers and practitioners to decide on which methodologies to seek to use in a specific situation based on knowledge about methodologies and information systems analysis and design.

Notice that the system does not exist in the real world, it is an explicitly formulated view-point. The main emergent property of the above root definition is 'that a decision is reached', a property that does only apply to the system as a whole and not to any of its parts. It is assumed in the above root definition that such a decision can be reached by the involved actors if knowledge about methodologies and information systems analysis and design is available.

Corresponding to the root definition is the conceptual model used in the survey. It corresponds in the sense that it is defensible against the root definition. The conceptual model contains the minimal set of necessary activities and their interdependencies to achieve what is expressed in the root definition. In this case, the minimal set consists of six activities, see Figure 3.1

1. This activity is necessary because the root definition says: "based on knowledge about ... information systems analysis and design", thus the knowledge must be obtained.

2. This activity is necessary for the same reason as for (1). It depends on (1) because the methodologies we look at here are thought of as being useful in information systems analysis and design.

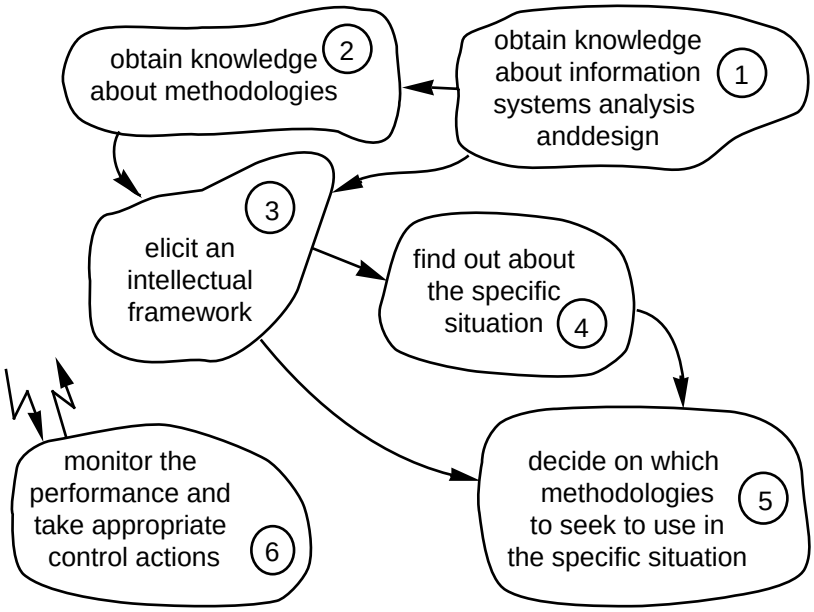


Figure 3.1: *A conceptual model used in the survey. It corresponds and is defensible against the above root definition. Key:* ◯*: activity;* $\rightarrow$*: dependency between two activities;* $\rightsquigarrow$*: dependency between an activity and all other activities.*

3. This activity is necessary in order to make the obtained knowledge into an intellectual form or framework that is relevant and useful in this particular system. Thus, it is highly dependent on (1) and (2).

4. This activity is necessary because the root definition says: "to use in a specific situation", therefore some finding out about what is special about a particular situation must be done. It depends on the outlook of the framework, that is, on the outcome of (3).

5. This activity is necessary for the obvious reason that, according to the root definition, a decision has to be taken. It depends on the outlook of the framework (3) and on what has been found out about the situation (4).

6. This activity is necessary to ensure that activities (1)–(5) are well performed by monitoring performance and taking appropriate control actions. Thus, in order to monitor all other activities it is de-

pendent on these, and the control actions taken changes activities (1)–(5) making them all dependent on (6).

It is worth noticing that the activities and the dependencies only describe *what* necessarily must be done, but not how to do it.

Each of the appreciation approaches will be mapped onto the conceptual model. In order to do this, it is necessary to distinguish between two roles: the researcher and the systems developer. The researcher has developed an appreciation approach and the systems developer is using it. The mapping of each of the appreciation approaches onto the model is done to see *how* each of the activities has been performed by the researcher while developing the approach or *how* they are to be performed by the systems developer while using the approach.

For each of the appreciation approaches the detailed mapping is done by asking the following questions for each activity in the model:

α. If it has already been done by the researcher as part of developing the appreciation approach:

1. How was it done?
2. What was achieved?

β. If it is left for the systems developer to do as part of using the approach:

1. Are there guidelines as to how it should be done?
2. What is the desired outcome?

These questions are interesting because it is common to all the approaches in the survey that they relate to the fundamental question 'Which methodologies in which situations?' one way or another. Of course, the researchers behind the approaches did not necessarily have this simple and pragmatic question in mind when they developed them; but the approaches can all be seen as though they seek to answer that question. In the following we will investigate to what extent and how well this question is answered in the literature. Since the conceptual model consists of the minimal set of activities necessary to answer the question this investigation is done by assessing which activities have been covered by the researcher (α-questions) and which have not yet been covered and therefore left for the systems developer to do (β-questions).

The Mapping

I have chosen to let the survey deal with eight appreciation approaches.[2]

Throughout this section the numbers, (x), refer to the activities in the conceptual model.

Taggart & Tharp

Taggart & Tharp have developed one of the earliest appreciation approaches found in the literature.[3] They have what readily could be called a management information requirement analysis view on information systems analysis and design, (1). On the basis of this view and on a literature study of known methodologies in 1977, (2), they derive a framework, (3).

The framework consists of four categories: development process, information, decision making, and organisation. For each of the categories there are a number of key aspects, e.g. for organisation: organisation environment, organisation subsystems, and management function and level. Each of the considered methodologies is assessed for each of the key aspects on a scale from 1 to 3 expressing the extent of treatment. 1 means 'aspect not considered', 2 means 'recognition given to aspect', and 3 means 'significant treatment of aspect'. For example, in organisation environment: does the methodology recognise that the simplicity or complexity of information needs depend on the stability of the organisation's external environment and internal structure.

I find Taggart & Tharp's approach to be based on much too simplistic measures in the sense that each aspect must be assessed on a scale from 1 to 3. It is also a problem that they claim the framework to be comprehensive.[4]

---

[2]Besides the eight approaches, the whole survey has included the three CRIS proceedings: (Olle *et al.* 1982; Olle *et al.* 1983; Olle *et al.* 1986). I have also examined (Shomenta *et al.* 1983; Benyon and Skidmore 1987; Yadav *et al.* 1988). The survey does not include appreciation approaches that look at methodologies *strictly* from the view-point of programming: (Bergland 1986) and (Davis 1988); and it does not include appreciation approaches that look at systems methodologies in general without relating to information systems, e.g. (Oliga 1988).

[3](Taggart and Tharp 1977).

[4](Taggart and Tharp 1977, p. 275). This may have been a reasonable statement in 1977, but what Taggart & Tharp deal with would today be called theories rather than methodologies.

Brandt

The three CRIS conferences (Comparative Review of Information Systems Design Methodologies) are sizable efforts in comparing methodologies.[5] The first conference was established around a test case concerning organisation of a conference (The IFIP Case) and each contributor was asked to apply a methodology to the test case. Each contribution was reviewed by a committee according to a large set of questions about methodologies in general and about how well they handled the IFIP Case in particular. The purpose was that of taking stock and presenting a spectrum of methodologies. The second conference had the purpose of feature analysis of the methodologies presented at the first conference. A number of approaches for assessing features came up. Brandt's approach was one of these.[6]

Knowledge about methodologies has been obtained by studying how the IFIP Case had been handled, (2). Brandt arrives at a list of features that she finds important, (3). The seven features are: origin and experience, development process, data model, iteration and tests, representation means, documentation, user orientation, and tools and automation prospects. Having set up the list of features a comparison of methodologies is made.

The CRIS effort is remarkable, but also worrying. More than anything it shows the difficulty in developing one common test case and then hoping that the methodologies under investigation are suitable for that case. For example, it is assumed in the IFIP Case that a computer system is going to be developed, its requirements are fixed, there are no users to interact with, etc. What if you have a methodology particularly suited for defining the problems of the organisation in a participatory process? In Brandt's appreciation approach I find that the comparison is very superficial in the sense that little clarity is provided to the question of what are the important and relevant difference between methodologies. This may have been caused by the CRIS setting, but it may just as well have been caused by the lack of referenced insight, theoretical or practical, into information systems analysis and design.

Wasserman, Freeman & Porcella

Wasserman *et al.* provide an overview of a great number of methodologies in order to see to what extent they can be integrated with the

[5](Olle *et al.* 1982; Olle *et al.* 1983; Olle *et al.* 1986).
[6](Brandt 1983).

programming language Ada.[7] Based on a theoretical exploration and reasoning from the viewpoint of information systems development as software development, (1), they have designed a questionnaire about methodologies. The knowledge about methodologies they have obtained stems from the questionnaires answered by those who developed each of the methodologies, (2). Twenty-four of the responses refer to methodologies actually in use and thus hope to draw upon experience with the use of the methodologies.

The intellectual framework behind their overview consists of six areas of concern, (3): life cycle coverage, applicability, technical concepts supported, work-products and representation, quality assurance, and usage. The overview of the twenty-four methodologies provides insight into how each of them deals with the six areas.

An obvious objection to this appreciation approach is that they rely fully and uncritically on experience gained by those who developed the methodologies. Wasserman *et al.* are aware of this, but it is almost like trusting a car sales-man about the value of a particular car in which you have showed some interest. A strength of this approach is, however, that they seek to gain the overview from an explicit and precise viewpoint, i.e. program development.

Floyd

Floyd has studied the application of a few methodologies in practice.[8] The methodologies have been studied, (2), by "teaching courses in which the method[ologie]s were presented, tried out by students on a case study and subsequently evaluated".[9] Based on a software engineering view on information systems development, (1), and the experience gained through laboratory experiments a framework has been elicited, (3).

Floyd offers a few concepts for categorisation of methodologies as an intellectual framework. The concepts are: area of application, perspective, guidelines, theory, coherence, coverage, and product-oriented features. The categorising concepts are not used very much by Floyd as she prefers the concrete experience with methodologies. A typical claim is: "the notion of 'action' [in JSD] is confusing since it has no time dimension".[10]

[7] (Wasserman *et al.* 1983).

[8] (Floyd 1984; Floyd 1986).

[9] (Floyd 1986, p. 20). Floyd too, use the term 'method' for what in this thesis is called 'methodology'.

[10] (Floyd 1986, p. 30).

A problem with Floyd's approach, I find, is that it lacks theoretical coherence in the framework. There is seemingly nothing to tie the characterising concepts together and thereby provide an argument as to why these concepts are useful. Floyd argues, to some extent, that her framework is intentionally vague in the direction of categorising and setting a taxonomy for methodologies as her purpose is more to clarify the area of methodologies as no suitable criteria for a taxonomy can be found. At this point, Floyd is using the obvious lack of an objective criterion as an argument for not being precise and consistent from a particular theoretical viewpoint. On the other hand, Floyd's investigation provides much substantial and useful knowledge about the actual use of methodologies.

Ciborra, Bracchi & Maggiolini

Ciborra *et al.* are most likely the first to seek to relate the differences of methodologies to differences of situations in information systems analysis and design.[11] As they put it: "different task environments imply different problem spaces and, above all, different method[ologie]s and approaches"[12]

Their appreciation approach is based on Newill & Simon's view of problem solving and theoretical work on information systems analysis and design, (1). The knowledge about methodologies has been gained through a literature study, (2). It is from this theoretical standpoint and with methodologies in mind that they derive their intellectual framework, (3).

In the framework, a situation, or task environment as they call it, is seen as consisting of four characteristics: organisation and information system, technology, users and systems developers, and, finally, project management. To each characteristic there is a measure. For example, for users and systems developers it is twofold: 'Are the analysts dependent on involvement with the users or can they work detached?' and 'Do the users act passively or actively?' Methodologies can be assessed according to what characteristics they can cope with.

As part of the approach it is claimed that it is possible to assess the characteristics of a specific situation, (4). From this assessment, an appropriate methodology is found, (5). The choice is simple since it is only necessary to find a methodology which can cope with the characteristics of the situation.

A weakness of this approach is, as I see it, that it is not at all clear

[11](Ciborra *et al.* 1980).
[12](Ciborra *et al.* 1980, p. 52).

how each of the characteristics should be measured, e.g. whether the users are active or passive may not always be measurable with sufficient significance. On the other hand, the characteristics provide some insight into the relationship between situations and methodologies in a simple and logical way.

Davis

Davis's approach[13] for matching methodologies and situations is probably the most well-known in its field. It is based on Simon's work on human information processing and Davis's application of this on information systems analysis and design, (1). A number of factors that influence the level of uncertainty are identified.

Knowledge about methodologies, (2), stems primarily from a literature study.[14] The framework is established, (3), as a match between the uncertainty factors and how much uncertainty a methodology can cope with. The match is a single scale of uncertainty. This means that each methodology can be categorised according to how much uncertainty it can cope with. At the lowest level of uncertainty the category is named 'asking', then in rising order: 'deriving from existing system', 'traditional analysis', and 'experimentation'. The methodologies mentioned in Section 2.1 all belong to traditional analysis.

The level of uncertainty of a specific situation is determined by a three step procedure, (4). First, the situation is characterised by organisational context, information system, users, and systems developers. Second, the uncertainty of the development process is evaluated in terms of existence of requirements, users' ability to express requirements, and systems developers' ability to understand requirements. Third, the overall uncertainty is evaluated. Thereafter, a primary and secondary category of methodologies is selected and within this one or more methodologies is selected, (5).

One of the weaknesses of Davis's approach is that it is difficult to evaluate the uncertainty factors and hence to make convincing evaluations. Another weakness is that it is not at all clear that uncertainty is sufficient as the one and only measure. On the other hand, the strength of the approach is that a systems developer in a particular situation will be forced to think about what undoubtedly are relevant aspects of the situation and by that find arguments for a decision on which methodologies to seek to use.

[13] (Davis 1982).

[14] Though an exception is a comparison of two methodologies based on practice (Munro and Davis 1977).

Episkopou & Wood-Harper

Wood-Harper & Fitzgerald developed a taxonomy of methodologies.[15] They claim that methodologies fall into six categories: general systems theory, human activity systems, participation, traditional, data analysis, and structured systems methodologies. (It is striking that the category general systems theory is without content.)

Later, Episkopou & Wood-Harper have taken the taxonomy and Checkland's work on problem systems and developed an appreciation approach they call a framework for choosing appropriate methodologies.[16] The approach is based on action research involving more than 60 systems developers.[17] Even if there is no explicit evidence it seems to be from these research efforts that the insight into information systems analysis and design and methodologies is gained, (1), (2).

The framework itself, (3), is structured according to Checkland's problem systems where distinction is made between problem owner, problem solver, problem contents system, and problem solving system. The approach establishes which factors it is important to consider when matching situations and methodologies. For the problem owner and problem solver, it is factors like cognitive style, skills, and ability to specify. For the problem content system, it is factors like resources and interest groups. As part of the framework, methodologies are classified according to the underlying ideology, tools provided, inquiring system, manpower and time needed.

In order to learn about a specific situation, (4), assessment of the problem owner, the problem solver, and the problem contents system is performed. After this, the problem solving system is formulated by choosing one or more methodologies, (5), that are appropriate relative to the assessment.

One of the weaknesses of Episkopou & Wood-Harper's approach is that they have not established a reasonable relationship between the situational factors and the features of methodologies. They are, of course, aware of this as well as the fact that the appreciation approach has not been tried in practice. A strength of the approach is the conceptual clarity of the areas to consider carefully, i.e. problem owner, problem solver, etc.

[15](Wood-Harper and Fitzgerald 1982).
[16](Episkopou and Wood-Harper 1986).
[17](Episkopou and Wood-Harper 1986, p. 277).

Jayaratna

Jayaratna's approach towards methodologies and situations is called Normative Information Model-based Systems Analysis and Design, NIMSAD for short.[18] Basically, it is a framework for understanding and evaluating methodologies and their use in information systems development.

NIMSAD has been developed and used over a period of years and experience has been gained with methodologies, information systems analysis and design, and NIMSAD itself, (1), (2), (3). The framework itself is a model of information systems development with eight stages.

1. Introduction to the 'real world'.
2. Understanding the situation of concern.
3. Diagnosis.
4. Prognosis outline.
5. Systems analysis.
6. Logical design.
7. Physical design.
8. Implementation.

Together with these eight stages a few methodologies have been evaluated by finding out how they support each of the stages.

When it comes to understanding a specific situation and deciding which methodology to use, (4), (5), NIMSAD is unusual. It is unusual in the sense that it is not a framework which provides guidelines for performing (4) and (5) once and for all in a project, e.g. where the project in established. On the contrary, it is a framework that is intended to guide the systems developer all the way through a project by providing opportunity 'at a conscious level of concern' to re-evaluate and re-select methodologies, (6).

The strength of Jayaratna's approach is twofold, at least. First, it has shown its usefulness through practice. That cannot be said by any other approach in this survey. Second, it is worth noticing that the approach has something to say about activity (6), i.e. the monitor and control activity. The approach encourages the systems developer to re-evaluation and re-selection which involves re-doing activities (2), (3), (4), (5). A weakness of the approach is that it is difficult to distinguish phases

[18] (Jayaratna 1986).

like diagnosis, prognosis outline, and systems analysis. In practice, they are seldom carried out in that order, but, moreover, they are likely to be inseparable in action and in logic.

Summary of the Mapping

The most significant differences between the appreciation approaches can be summarised by clustering the mapping onto activities (1), (2), (3) and (4), (5), see Figure 3.2. This leaves out activity (6), but only Jayaratna deals with that anyway.

The summary in Figure 3.2 reveals four categories of appreciation approaches, see Figure 3.3. The theory-based approaches are based on theoretical research efforts in activities (1), (2) and (3) while the practice-based are based on practical research efforts. The approaches that deal with methodologies alone cannot be mapped onto activities (4) and (5) while the approaches that deal with both methodologies and situations have something to say about how to do (4) and (5).

It is a characteristic of all the approaches found in the literature that they are stable and general.[19] They are stable in the sense that the frameworks are unadaptive. They are general in the sense that unique situational knowledge is disregarded. That is not necessarily a quality. What if a new methodology is developed and its features cannot be captured by means of a stable appreciation approach? What if information systems analysis and design changes likewise? What if systems developers in a particular situation possess substantial knowledge about methodologies that they want to apply systematically and this is impossible by means of a general appreciation approach? I shall argue that there are indeed good reasons for developing appreciation approaches which are dynamic rather than stable and situational rather than general.

3.2 From General Approaches to Situational Approaches

Schön has made a thorough analysis of how practitioners think and act.[20] The analysis is about the thinking and acting of professional practitioners in general. It can, however, easily be argued that systems developers can be seen as professional practitioners, too.

[19] The only exception being (Jayaratna 1986).
[20] (Schön 1983).

	(1), (2), (3): obtain knowledge about ISA&D and methodologies and elicit framework	(4), (5): find out about specific situation and decide on methodologies	Strengths and weaknesses
Taggart & Tharp	A management view on ISA&D and literature study of methodologies → a survey of methodologies	n/a	(−): simplistic measures
Brandt	A list of features of methodologies and the IFIP Case → a comparison of methodologies	n/a	(−): superficial comparison and the artificiality of The IFIP Case
Wasserman *et al.*	A programming view on ISA&D and questionnaires about methodologies → an overview of methodologies	n/a	(+): explicit and precise viewpoint behind overview (−): rely on authors' own claims
Floyd	Laboratory experiments with methodologies → experience with methodologies	n/a	(+): substantial insight (−): too little coherence in framework
Ciborra *et al.*	A Simonean view on ISA&D and literature study of methodologies → a match of task environment characteristics with methodologies	Determine characteristics of present situation	(+): a simple relationship between situations and methodologies (−): unclear measures
Davis	A Simonean view on ISA&D and literature study of methodologies → a match of uncertainties of situations with methodologies	Determine uncertainty of present situation	(+): forced to think through relevant aspects (−): difficult to believe in uncertainty measure
Episkopou & Wood-Harper	Action research → factors of importance	Assess problem owner, problem solver, problem content system and formulate problem solving system	(+): conceptual clarity (−): no relationship between situations and methodologies
Jayaratna	Used in practice → a model of ISA&D relevant to understand methodological practice	(Re-)evaluate and (re-)select methodologies throughout a project	(+): it is useful and the framework can be reworked from within (−): difficult to distinguish phases

Figure 3.2: *The mapping of the appreciation approaches in summary. Key: →: leads to; n/a: not applicable.*

	Theory-based	Practice-based
Methodologies	(Taggart & Tharp 1977) (Brandt 1983)	(Wasserman *et al.* 1983) (Floyd 1984 & 1986)
Methodologies and situations	(Ciborra *et al.* 1980) (Davis 1982)	(Episkopou & Wood-Harper 1986) (Jayaratna 1986)

Figure 3.3: *Four categories of appreciation approaches.*

Schön distinguishes between two different modes of thinking: technical rationality and reflection-in-action. Technical rationality is seen as instrumental problem solving. The practitioner takes a goal as given and by selecting the best means seeks to reach this goal.[21] Hence, technical rationality is based on the assumption that there is consensus about the goal and that the goal is visible and clear. The selection of best means is done by applying the relevant scientific theory.[22] Knowledge is mainly a result of science and can be separated from practice, where professional knowledge is seen as three types: an underlying basic science, an applied science, and skills and attitudes. In this sense practice and science (research) remain separate where the knowledge produced by research is specialised and standardised theories and techniques that can be applied by practitioners to diagnose situations and solve problems. Practical knowledge is knowledge about the relationship of means to ends.

In contrast to technical rationality, reflection-in-action is seen as problem setting where each situation is considered unique: "the practitioner approaches the practice problem as a unique case ... seeks to discover the particular features of his problematic situation, and from their gradual discovery, design an intervention".[23] The three most significant differences between technical rationality and reflection-in-action are shown in Figure 3.4.

Schön discusses why technical rationality must be abandoned and why we must see professional practice as reflection-in-action. Firstly, a situation is fundamentally unique:

> Even when a problem has been constructed, it may escape the

[21](Schön 1983, p. 21ff).
[22](Schön 1983, p. 34).
[23](Schön 1983, p. 134).

	Technical rationality	Reflection-in-action
Situations	Falls into scientifically defined categories	Are unique, complex, uncertain, and value-conflictual
Knowledge	Is a result of science and must be separated from practice	Is inseparable from action
Practice	Is fundamentally different from research (science); practice is application of theory and research is production of theory	Includes research

Figure 3.4: *Significant differences between the two view-points on professional practice.*

> categories of applied science because it presents itself as unique or unstable. In order to solve a problem by the application of existing theory or technique, a practitioner must be able to map those categories onto features of the practice situation. ... But a unique case falls outside the categories of applied theory; an unstable situation slips out from under them.[24]

Secondly, taking a goal as given ignores problem setting. Schön argues that in practice problems are not given, they are constructed from the problematic situation. The process of setting the problem is:

> a process in which, interactively, we *name* the things to which we will attend and *frame* the context in which we will attend to them.[25]

That is, ends are not given, ends and means are mutually dependent and are therefore set in the same process.

Let us now return to the appreciation approaches with Schön's theory in mind. The approaches in the survey all take the stance of technical rationality to some extent. All the approaches define a stable framework where the concepts that distinguish methodologies are found

[24] (Schön 1983, p. 41).
[25] (Schön 1983, p. 40).

during research and never changed later on. In this sense they rely on the assumption that knowledge is a result of research and that systems developers involved in practice merely apply the knowledge already established. That is, they assume a strong division between research and practice. The approaches that deal with both situations and methodologies all take the stance where situations and methodologies can be categorised. This means, in terms of Schön, that situations fall into well-defined categories and having determined to which category a specific situation belongs, the optimal methodology can be chosen. Thus, the approaches are general in the sense that the uniqueness of a situation is disregarded. It is in this sense that I claim the surveyed approaches belong to the technical rationality tradition. Jayaratna's approach is a slight exception from this because it allows and almost encourages the systems developer to modify parts of the framework and hence move in the direction of reflection-in-action.

Schön would very much like to abandon the whole idea of technical rationality. I shall, however, merely claim that the appreciation approaches belonging to technical rationality are insufficient. They are useful but insufficient. They are useful because it is possible and useful to investigate the features of methodologies and situations. If, for example, experience from practice shows that DeMarco's data-flow technique is not useful when describing issue-based work in a bank, because it is not in logic nor in practice routines, then this experience is generalisable into knowledge about data-flow detached from the situation where it was experienced.

On the other hand, the approaches are insufficient because the general argument put forward by Schön does apply to the use of methodologies in information systems analysis and design. From this point of view we must strive at practising reflection-in-action in the process of matching methodologies and situations. The consequence is that appreciation approaches must *also* take into account that situations are unique and substantial knowledge is gained through practice by systems developers. This calls for appreciation approaches which are more dynamic and situational. The work ahead lies in finding out how to do the *dynamic and situational thinking* without losing the general methodological knowledge of existing approaches.

The idea I will bring forward here is to base such an approach on SSM.[26] SSM is a systems approach for problem solving or, more correctly, for learning in unstructured situations. It is close to the process of

[26](Checkland 1981).

reflection-in-action in the sense that when using it situations are seen as unique and insight into the uniqueness is gained.

The starting point of SSM is an unstructured problematic situation. The problems perceived are not taken as given, but rather as a rich expression of the nature of the situation sought. Having expressed the richness, several root definitions are formulated as systems views that are potentially relevant to learn about the situation. The root definitions denote views on the situation and are not accounts of the real world as it is. For each root definition a conceptual model is built. The conceptual models are then compared with the situation. Based on the comparison of several conceptual models with the situation feasible and desirable changes are found. The conceptual models structure a debate amongst the actors. By this the actors define for themselves which changes to bring into action.

Now, if the problematic situation has to do with finding out which methodology to seek to use we can use SSM. Relevant views on the situation are modelled as systems views. Within this, it is possible to take the existing general methodological knowledge to be views of the situation. The methodological knowledge possessed by actors in the situation is taken to be views as well. Formulating root definitions and building conceptual models of these general and local (situational) views enable a comparison and debate about which methodologies to seek to use.

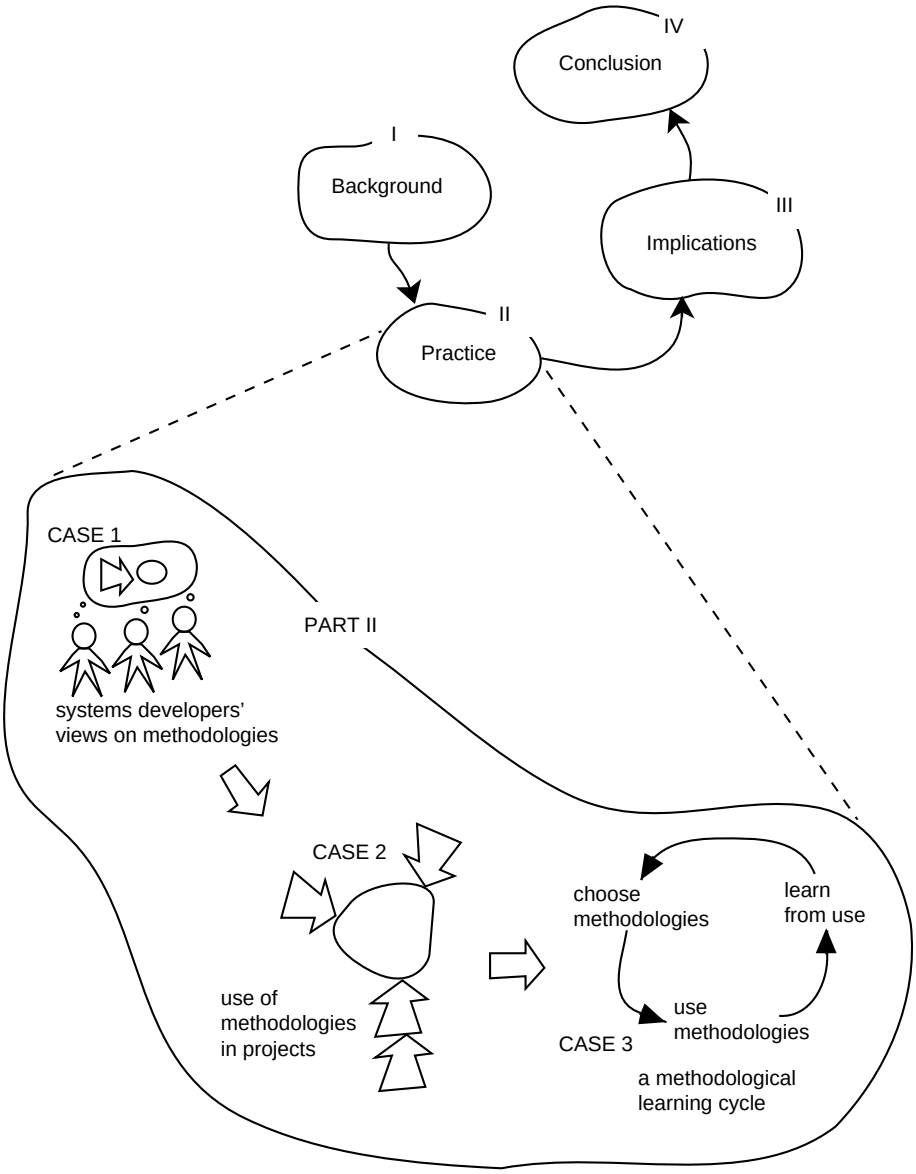
3.3 Summary

I started out in this chapter stating explicitly the standpoint from which I wanted to survey the literature on appreciation approaches. The standpoint was formulated as a root definition and detailed as a conceptual model. The survey was then made by mapping each of the approaches onto the conceptual model. This highlighted to what extent and in what way each of the approaches has answered the question: Which methodologies in which situations? It also illustrated some of the limitations and qualities of each of the approaches.

By means of Schön's work on reflection-in-action I argued that the surveyed approaches, except Jayaratna's, have taken the stance of technical rationality in the sense that they are stable and general. I also argued that this stance is insufficient and that there is a need for appreciation approaches that take the stance of reflection-in-action in the sense that they should be dynamic and situational. This should be done without abandoning technical rationality, but by utilising the best of it.

By means of Checkland's work I suggested that one place to search for such an approach is within soft systems thinking. Within this theoretical and practical tradition we may simultaneously utilise existing knowledge about methodologies and situations and establish new methodological insight through a process of dynamic and situational thinking.

Part II
Practice


IV
Conclusion
I
Background
III
Implications
II
Practice
CASE 1
PART II
systems developers'
views on methodologies
CASE 2
use of
methodologies
in projects
choose
methodologies
learn
from use
use
methodologies
CASE 3
a methodological
learning cycle

4

The Individual Level

This chapter deals with the use and understanding of methodologies at the most basic level: the individual level. The empirical basis, Case 1, has been established by interviewing a few systems developers who use methodologies in their daily work. Case 1, which stems from the Data Processing Department of the Provident Merchant Bank, provides insight into the methodological reasoning at this level.

Section 4.1 is about the overall design of the inquiry. The inquiry was set up to learn about methodologies at the individual level. It includes the research themes and a brief discussion of qualitative interviewing as a research approach. Section 4.2 presents interviews with three systems developers about their relationship with methodologies. In Section 4.3 the diversity of attitudes towards methodologies amongst systems developers is discussed. This is done by focusing on the relationship between, on the one hand, the experience and the values of the individuals and, on the other hand, their use of methodologies. Finally, Section 4.4 provides a short summary.

4.1 The Inquiry

The focus in this chapter is on the relationship between the individual systems developer and the use of methodologies. To appreciate this, it is necessary to be explicit about the research themes and setting and about the research approach.

Themes and Setting

The purpose of the inquiry was to get insight into how systems developers view their own use of methodologies.

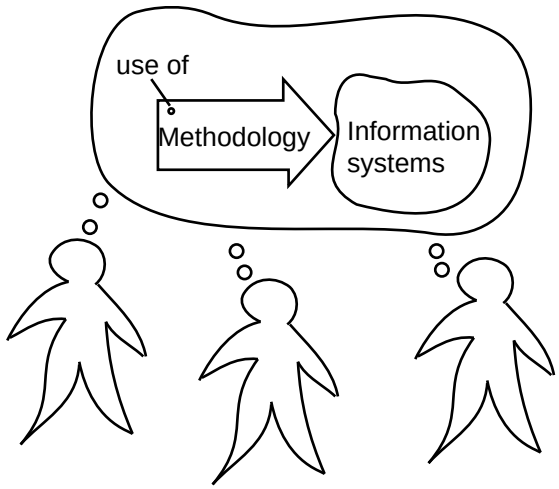


Figure 4.1: *Case 1: The individual level. What are systems developers' views on the use of methodologies?*

Figure 4.1 expresses the focus of the inquiry. The overall research theme can be expressed as follows:

Individuals and methodologies: How are methodologies claimed to be used by systems developers? How do systems developers view methodologies?

Methodological reasoning: What methodological reasoning applies at the individual level?

The inquiry took place in the DP Department of the Provident Merchant Bank. The department employs a hundred systems developers. They develop all information systems in the bank. They are organised in groups with responsibility for each their part of the information systems, e.g. basic customer information and customer accounts, arbitrage, and accounting. The department trained many of the systems developers during the last five years in the use of a few methodologies and a variety of simple techniques and standards.[1] The systems developers

[1]The issue of learning and introducing new methodologies in the DP Department of the Provident forms the basis of Case 3, see Chapter 6.

have learned these methodologies to varying degrees of excellence and apply them with varying degrees of confidence.

To investigate the above themes a number of systems developers in the Provident were interviewed. The themes were made operational by formulating the list of questions below. The questions were not asked directly and as phrased here; but they did structure the interviews in the sense that they were covered one way or another during the interviews. The interviews were informal in the sense that the interviewees were asked to tell what they found to be relevant and important regarding methodologies. They never saw the list of questions and it was my task as interviewer to guide the talk through the questions.

- What is your background within information systems development?
- What are you currently working on?
- Which methodologies do you use?
- With what purpose do you use them?
- What have you gained/not gained from this?
- Do you use methodologies 'by-the-book'? Why? Why not?
- Which methodologies do you know but do not use? Why?
- What is your general impression of methodologies?
- What importance do you assign to the use of methodologies?

These questions intentionally guide the research in the direction of personal opinions and viewpoints.

Qualitative Interviewing as a Research Approach

The difficulty of doing research into the individual use of methodologies is to get insight into what the individuals 'really' think about methodologies. Qualitative interviewing was chosen as the main research approach in order to deal effectively with these difficulties.

Research based on qualitative interviews and other qualitative techniques has been described and discussed by Patton.[2] To Patton, qualitative research techniques are alternatives to quantitative techniques. One

[2](Patton 1980). Many others have dealt with qualitative research, but Patton's work is useful and sufficient for describing the research reported in this chapter.

of the major differences between qualitative and quantitative techniques are the number of persons (or other units of analysis) considered and the ways of interpretation.

> Qualitative data provide *depth* and *detail*. Depth and detail emerge through direct quotation and careful description.[3]

This conforms very well with the basic ideas of action research. Here, as in action research in general, we have one or very few cases to draw upon. There is not a standardised way of generalisation and each lesson learned and each conclusion reached must be supported by context and reasons as for others to judge for themselves about their usefulness.

Qualitative interviewing is a relevant research approach from the point of view of the research themes. Patton writes:

> The fundamental principle of qualitative interviewing is to provide a framework within which respondents can express their own understandings in their own terms.[4]

This means that we can *enter* a situation, get substantial insight, and develop an insider's view of the research themes. Thus, qualitative interviews may give some insight into what the systems developers 'really' think about methodologies, their attitudes, and, in general, their relationship with methodologies.

There are three archetypes of interviewing forms where the difference has to do with the degree to which questions are determined before the interviews.[5] 'Informal conversation' is spontaneous generation of questions as a natural part of an interaction. 'General interview guide' is a set of issues which it is relevant to cover in talking to each interviewee, but they need not to be asked in a standardised way or sequence. 'Standardised open-ended interview' is a set of predetermined questions where each interviewee is taken through the same series of questions.

In the interviews with the systems developers a mixture of informal conversation and general interview guide was applied. The themes stated above should be covered by the interviews, but many of of the important issues were not anticipated, they emerged during the interviews. The issues of 'experience' and 'professionalism' emerged during the interviews and ended up being among the major findings.

[3](Patton 1980, p. 22).
[4](Patton 1980, p. 205)
[5](Patton 1980, p. 197–205).

4.2 Three Systems Developers

As part of designing the interviews several systems developers, managers and staff members were asked about who should be interviewed in order to get a broad and detailed view of the department. On this basis eleven interviewees were selected by a group of staff called the Methodology Group. Based on these eleven interviews I have selected three to be presented here. The selection is based on the criterion that they happened to portray three different and prominent attitudes towards methodologies.

The three interviewees[6] are presented in the following. In order to provide anonymity they are called: the Experienced, the Confident, and the Professional. The names have been chosen to designate the main characteristic of each of them.

The Experienced

The Experienced has worked as a systems developer in the Provident for several years. Before that he was a bank clerk in the Provident. He was transferred to the DP Department when it was still very small and insignificant and he is now the leader of the Marketing Group. The Marketing Group develops and maintains all computer-based information systems used in marketing, i.e. both information systems for analysis of customers and information systems for promotion of new products. He has received all his training as a systems developer in the department and during his time there he has participated in many projects of varying nature.

As leader of the Marketing Group his primary task is to negotiate with the users which new projects to start and what the remit should be.

> "I'm the main contact between our group and the users. ... and at the meetings where we decide what new project to start I draw simple activity diagrams and I keep detailing them until I'm sure they understand them."

This kind of meeting characterises to a large extent the way he uses methodologies. He does not use the methodologies as prescribed, but, rather, he likes to take individual techniques from a methodology and use them in his own way without too much consideration about the original intentions.

[6]The interviews were documented in *Field Notes and Protocols*, (Nielsen 1988b). The citations in this section stem from these protocols.

> "I know a methodology for modelling data, their structure and interrelation, but I don't really use it as prescribed. Same thing with data-flow models."

As an example, he uses the principles from data-flow models when making activity diagrams, but he does it without balancing input and output and without describing data in data dictionaries. He does this well-knowing that these techniques are considered important in most textbooks on data-flow modelling.

The Experienced never uses methodologies as prescribed. It is fair to ask: Why? To this he replies that during the years that he has been with the department he has tried a few methodologies and a great number of techniques. None of these have so far been able to transcend the common sense of an experienced systems developer. His experience tells him that one should be very careful when using a methodology because it creates a feeling of 'doing the right thing'. Experience, he argues, is when you know when to use a particular technique.

On the other hand, the Experienced is very much aware of the need for methodologies.

> "The data in our systems are much more complex than just a few years ago. And with DB2[7] it is necessary to model data much more carefully."

The Experienced finds that despite the fact that he is very experienced he will have to use methodologies more extensively in the future. In general, the Experienced finds that methodologies are becoming more and more useful and necessary. Methodologies are improved and the information in the bank becomes more complex and so does the analysis and design of information systems.

The Confident

The Confident has been with the DP Department for three years and before that he received two years of basic training and education in programming. He is a member of the Accounting Group, a group of 6 systems developers responsible for all computer-based information systems to do with the area of accounting both internally and externally. During the three years he has mainly been working as information analyst and programmer in small projects. In the latest project, a project group of three persons developed over a period of five months a new

[7]DB2 is a relational database system on one of the most popular mainframe architectures.

information system for generating annual reports. The project was initiated as a response to the Government passing a new act on annual reports for banking. Lately, he has begun to spend more and more of his time on maintenance. Maintenance involves correction of errors and minor adjustments of the information systems for which his group are responsible. He feels very comfortable with this.

The Confident seldom relies on methodologies in his work. In a typical project he is given a task by the group leader. The description of the task states the overall purpose of the system to be developed. In the beginning of a project the analysis and design of the systems-to-be is done by talking informally with the users.

> "At the meetings with the users we are talking. We don't use any particular methodology."

In the Accounting Group, it is left to each of the systems developers to decide which methodologies to use. In most cases, the Confident will communicate with the users in terms of structures, field names in the existing database, and screen lay-outs.

In his mind, it is not necessary to apply any systematic way of describing the information systems and eliciting the needs for computer support. He believes that it is unnecessary to analyse by means of methodologies because what he is already doing seems to be sufficient. The users in the bank with whom he is working are more or less used to having that kind of meeting.

> "Our users are computer-minded."

If methodologies are not used to any significant extent how can he then know whether he has understood the tasks that need to be supported by the information systems he is developing?

> "Basically, we know what the users need."

The last statement shows the confidence that he has in what he is doing. Why should he then use a methodology? Using a methodology is unnecessarily time-consuming because it forces him to make descriptions that are formal, systematic and detailed, and it does not help him in doing his work better or more effectively.

In more complex situations than those which he has experienced until now the Confident expects that he will develop information systems in very much the same way as he has done until now. He believes that his craftsmanship in getting the programs and databases right is sufficient. The Confident's attitudes towards methodologies are shared by many of his colleagues.

The Professional

The Professional has worked in the DP Department of The Provident for more than four years. Before coming to the department he took a one year course in programming and systems analysis. He is a member of the Credit Group. The Credit Group is responsible for the development and maintenance of computer-based information systems related to loan business and other types of credit given to customers. He has participated in a few projects within his group and has recently been promoted to project leader. Currently, he is leading a project with five systems developers including himself. The purpose of the project is to develop a new information system for calculating, retrieving, and presenting balances of credit accounts.

The Professional has for a long time had a keen interest in methodologies and their use in his work.

> "I try to use methodologies as much as possible. And I encourage my colleagues to do the same."

He is currently using DeMarco's SA/SD, a minor part of IM, and in an earlier project he experimented with a combination of SA/SD and prototyping. He tends to rely very much on methodologies and in doing that he attempts to follow the guidelines as closely as possible. This does not mean that the Professional is acting blindfolded with regard to methodologies.

> "I don't think SA/SD can be used as DeMarco advocates. For example, many of the decisions during analysis should be postponed to design."

Relevant questions become ones such as, why he relies on methodologies, what his expectations are and to what extent they are met? To answer this he speaks of IM, which he sees as a methodology with great prospects for improving a systems developer's understanding of information systems.

> "Modelling data properly is a matter of understanding better. ... It isn't because it's DB2 database technology."

As opposed to the Experienced, his motivation for using methodologies has little to do with the increasing complexity of the technology. It is a structured and systematic way of understanding the information systems. But it is more than that, it is also a way of imposing professional standards on information systems analysis and design.

> "[It is] difficult to see what [information] systems development really is. Systems developers like to go their own ways... but methodologies are important when we want to know what we are doing and why."

The use of methodologies is, to him, a way of having a conscious relationship with working practice, a way of talking about what to do. It is also a way of getting and maintaining high professional standards.

A few of the Professional's colleagues share his attitudes towards methodologies. It is characteristic that he often has to defend his viewpoints against attacks from the more experienced or the more confident systems developers in the Provident.

4.3 The Diversity of Individual Use

I argued theoretically in Section 2.3 that methodologies are distinct from working practice. The distinction arises because a methodology is an intellectual construct that may or may not inform a systems developer about what to do and how to do it. On the other hand, working practice is — practice. Practice is made up of the activities actually performed. The interviews with systems developers confirm the usefulness of this theoretical standpoint.

Lesson 1 *Methodologies are never simply used. They are adapted to a particular situation and the guidelines of a methodology remain different from the working practice.*

The Experienced is an example of a type of systems developer that adapts a methodology to his specific needs in a particular situation. His main activity is meetings with the users. At these meetings he finds it unnecessary to use a methodology in full. For example, he uses SA/SD when modelling processes to which information systems are to be developed; but his intentions are to get the users to talk about and be precise about 'what' they do and not as much whether data input can be balanced with data output of the modelled processes. In this way, his intentions are different from at least the main intentions behind SA/SD. Nevertheless, a part of the methodology is useful to him when it is adapted to the situation at hand.

The Professional, who is much more strict on how and when to use methodologies still adapts them to his present needs in the situation at hand. He is the most likely type of systems developer to seek to apply a methodology exactly as the guidelines advocate. Nevertheless, he does

not take a methodology to be a procedure or a recipe which can ensure a proper result if followed faithfully and undistorted. Rather, he strives at understanding the applicability and the consequences of a methodology. From this he decides how to seek to apply it. This is well illustrated by his remark about SA/SD: It cannot be used as advocated. Thus, it must be twisted to fit the situation, i.e. adapted.

Methodologies never seem to be used in a simple or straightforward way. More likely, they are adapted to a particular situation, both in the sense of only being used partly, like the Experienced does, and in the sense of being used with variations, like the Professional does.

The main impression from the interviews is that there is a significant diversity of individual use of methodologies. This is in a way not surprising considering Lesson 1. The following lesson should therefore be seen as a qualification of the above lesson.

Lesson 2 *There is significant variation in the degree to which different systems developers rely on methodologies. In one dimension this ranges from attempts to apply methodologies whenever possible to reluctance to use even the most straightforward ideas, tools and techniques of a methodology. In another dimension it ranges from attempts to apply methodologies literally to free interpretation of selected elements of a methodology.*

The Confident is a representative of the attitude amongst systems developers that methodologies are seldom useful. His attitude towards methodologies is common to many systems developers. It would certainly be too hasty to claim that his reluctance stems from negligence; rather it is based on the evaluation that methodologies are not sufficiently useful. He has never been in a situation where he failed or felt incompetent in a way that can be explained by the fact that he does not use methodologies.

The Experienced is somewhat sceptical towards methodologies. He uses methodologies on a small scale. Or, rather, he uses parts of methodologies. The activity models he is drawing at the meetings with the users are inspired by SA/SD, but he has no intention of using the formal way of building the models as DeMarco prescribes. Without considering the original intentions behind the methodology he has picked a few techniques out of their original context. In this sense he is a representative of the systems developers that interpret any part of a methodology with considerable freedom.

The Professional sees methodologies as one of the most powerful ways of professionalising information systems analysis and design. This is his

main reason for attempting to apply methodologies whenever he gets an opportunity. As he says in the interview: he seeks to use methodologies as much as possible. To the Professional, methodologies enable him to think and talk about what he is doing and why he is doing it. Professionalism is also his main reason for applying methodologies as literally as possible.

THE methodological reasoning at the individual level is mainly characterised by the influence of experience and professional values on the use of methodologies.

Lesson 3 *The way a systems developer uses methodologies reflects their experience and values. A systems developer with no or little experience is more likely to rely on methodologies than the experienced. A professional systems developer is more likely to find methodologies useful and necessary than the unprofessional.*

The Confident is inexperienced and does not share many of the values of the professional systems developers. He has only worked in information systems analysis and design for a few years. During this period he has mainly worked in small and similar projects. He is inexperienced, and he is also unprofessional in the sense that he does not believe it to be important to know what to do and why nor that high standards are necessary. On the contrary, he is rather confident that he is already doing the right thing; he does not give much thought to whether he could improve his working practice or not. The Confident, as an inexperienced systems developer, does not rely on methodologies since he never uses them. Additionally, as an unprofessional, he does not think that methodologies are useful or necessary, not even in more complex situations.

The Experienced is highly competent and he shares the values of the professional systems developers to some extent. It is, however, his great experience that seems to be primary. He wants to use methodologies and he uses them on a small scale. But, presently, there is not a real need for him to use methodologies. There may be several reasons for this. One reason is that during the past years he has internalised parts of the methodologies to such an extent that they are impossible to recognise in his working practice and that he is not aware of the influence methodologies have had on him. Another reason may be that because he is experienced he is less in need of ways of thinking and talking about what to do and why.

The Professional is not very experienced, but he is strongly in favour of professionalism amongst systems developers. He is also the most

prominent advocator of methodologies in the Provident. Methodologies, to him, are useful and necessary and he relies on them to a great extent.

4.4 Summary

When we look at the relationship between the systems developer as an individual and the use of methodologies the three lessons may contribute to our understanding of methodological reasoning. The methodological reasoning at the individual level consists of the arguments and the attitudes that make a difference to a systems developer when deciding what to do with respect to methodologies.

Lesson 1 tells us that what the systems developers actually do, i.e. the working practice, may be guided by a methodology, but it is not in a simple way the use of a methodology. A methodology will always be adapted in the sense that it is used with great variation and sometimes only partly. It is thus not surprising in Lesson 2 to learn that there are significant variations in the use of methodologies. The variations are along two dimensions: 'reluctance to use' versus 'whenever possible', and 'literal use' versus 'free interpretation'. Lesson 3 explains this diversity from the viewpoint that the systems developer's experience and values with respect to professionalism is reflected in the extent to which methodologies are considered reliable, useful and necessary.

Some of the insight gained can be summarised in the following contradiction.

Contradiction 1 *The main methodological contradiction at the individual level is between, on the one hand, the systems developer relying on experience, skills and background to appear implicitly and ad hoc and, on the other hand, the systems developer relying on explicitly stated methodologies and values.*

The contradiction expresses a limitation of methodological reasoning at the individual level. The contradiction both implies a choice or a balance that each systems developer has made or will have to make. The choice is a choice between the two aspects of the contradiction. The contradiction also expresses that the limitation will be there irrespective of whether the systems developer has made a conscious choice or not.

5

The Project Level

This chapter is about Case 2, an inquiry into methodologies used in information systems projects in The Royal Bank of Finance. The course of action in three similar projects is presented and reflected upon.

In Section 5.1, the research themes and research approach of the inquiry in the Royal Bank are outlined and discussed. After this three sections follow each describing a project. Section 5.2 is about a project using SA/SD as the one and only methodology. Section 5.3 is about a project using a specific organisation theory, i.e. Transaction Cost as suggested by Ciborra, combined with basic computer science skills. Section 5.4 is about a third project using a combination of JSD and Newman's models of office work. Section 5.5 reflects on the learning and use of methodologies at the project level.

5.1 The Inquiry

The inquiry in the Royal Bank is, on the one hand, an almost laboratory-like experimentation with methodologies and, on the other hand, authentic information systems projects. This makes it relevant to be explicit about the setting, the research themes of the inquiry, and the research approach.

Themes and Setting

Initially, the overall purpose of the inquiry was to evaluate and compare methodologies based on their practical use in information systems analysis and design.

The inquiry took place in The Royal Bank of Finance as a cooperative project between the bank and a university research group of seven masters students and their supervisor. The research group was divided into three separate information systems development projects with the same task but using three different approaches in three similar branches of the bank.

As expressed in the remit, the task the three project groups were facing in each their branch was:

> The task addresses the parts of loaning business that have to do with private (non-business) loans, building loans, re-mortgaging, and student loans. A description is wanted of how these activities are carried out today together with proposals for establishing integrated, computer-based support. The proposals for solutions must build on existing computer systems and be designed to use personal computers for word processing, document handling, archiving, electronic mail, information retrieval, and information processing, for example by means of simulation.[1]

Each project group was given three months to fulfill their task.

The remit expresses a strong commitment to analysis and to making descriptions of existing work as a foundation for design. The clerical workers had the most substantial knowledge about the work with loans. Because of this, it was decided to apply an overall participatory strategy where the research groups acting as systems developers were to cooperate closely with bank clerks. This meant that each of the three project groups were to cooperate with the clerical workers in their branch in the day-to-day analysis and design and only occasionally with the DP department of the bank.

The three branches were to a large extent similar. They were located in the same city and the districts they covered had very much the same kind of customers. The tasks performed in the branches were very similar encompassing the handling of: saving accounts, small loans and mortgages, and a few large loans for medium-sized businesses.

The participants in the research groups had common skills and background as masters students in their final year of computer science. Besides the traditional courses over the last four years in mathematics, programming, formal specification and reasoning, and basic information systems development they had all specialised in information systems development in their fifth and final year. Courses were taken in systems analysis and design (SA/SD and JSD in particular), organisation theory, and theory of knowledge as a major part of the specialisation.

[1] *Project Contract, The Royal Case.* (Mathiassen and others 1984).

The major difference between the three projects was the selected approaches:

North Branch: In this project SA/SD was selected as the only approach.[2] See Section 5.2 for a description of this project.

East Branch: Based on the group members' practical computer science skills this project took the Transaction Cost Theory (TC), a general theory of organisation and information, as their framework for understanding the bank and its need for information systems.[3] See Section 5.3 for a description of this project.

South Branch: This project chose to use JSD as their main design methodology.[4] Because the group believed that JSD could not cover the analysis part of the task they also chose to use Newman's Models of Office Work (MOW) as their main methodology of analysis.[5] See Section 5.4 for a description of this project.

The projects were independent in the sense that the members of the research group was a member of only one project and as the three projects ran simultaneously it was agreed not to have any interaction between the projects.

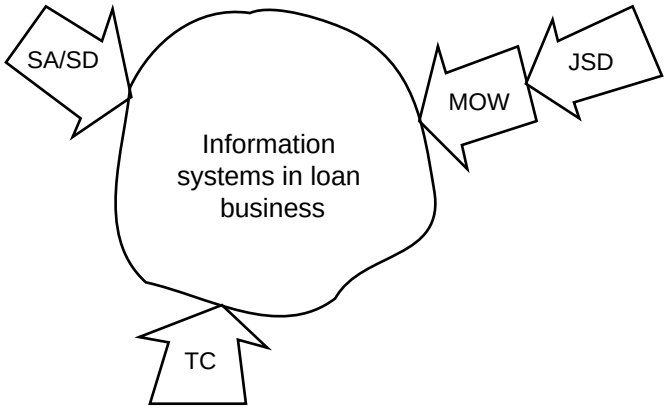


Figure 5.1: *Case 2: The project level.*

[2] (DeMarco 1978).
[3] (Ciborra 1981).
[4] (Jackson 1983).
[5] (Newman 1983).

The setting in The Royal Bank of Finance was realistic in the sense that the task and its uncertainty, the limited resources, the bank's demand for results, the cooperation between different interest groups all were highly realistic measures. These are important features of the projects and they add to its usefulness as a basis for research. However, the most significant feature is that the projects are authentic.

The initial purpose of evaluating and comparing the methodologies introduced three research themes:

The working practice: What is actually done to develop an information system? Why is it done and in what way is it influenced by the methodologies?

The use of methodologies: How are the methodologies used? What happens when the project seeks to use a specific methodology and, in particular, how does the frame of action and the modelling language support the work to be done?

Communication between users and systems developers: How is the communication performed? Especially, do the methodologies support the users' understanding of the information system and the to-be organisation and do they support the systems developers' understanding of the users' work and information needs?

These themes are tied to the evaluation and comparison of methodologies with respect to participation in an authentic setting. The themes express an intentional close relation to and interest in what is actually done.

In retrospect, it is apparent that the three projects can be used for more than a comparison of methodologies. The three projects can be seen as three different ways of bringing past experience and theoretical background into a new project. From this viewpoint it is interesting to inquire into the experience gained from using methodologies at the project level. This level is distinct from the individual level as it involves a group of systems developers and their *joint* effort in analysis and design of information systems. This includes a cooperative effort on what methodologies to seek to use and how to do it. The following general research theme is applied:

Methodologies in a project: How are methodologies used in a project? What methodological reasoning applies to the project level?

This theme is in many respects more general than the three above. It cuts across the three projects and it is not closely tied to the working practice.

The projects described in the following three sections illustrate the above four themes in different ways.

Diary Writing as a Research Approach

The setting of three separate but similar projects with the intention of evaluating and comparing the use of methodologies afterwards introduces the problem of how to:

- Document the working practice in each of the projects.
- Reflect on each of the approaches while in the midst of the projects.
- Document the insight gained and convey it afterwards.

Diaries were used as a means to achieve this.

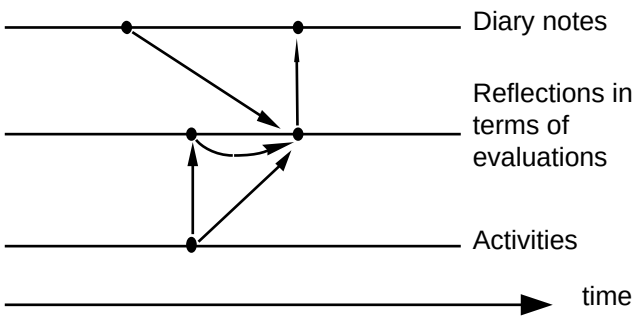


Figure 5.2: *Using diaries as a medium for evaluating activities. Evaluations are based on activities, previous reflections and diary notes and result in new diary notes. (Jepsen* et al. *1989, p. 211)*

The idea of using diaries stems from Naur who uses diaries as an empirical research technique in his studies of the process of programming.[6] His idea is to record actions and thoughts, as accurately as possible, while in the midst of them. In accordance with Naur's idea we wrote diaries in each of the three projects.

Diaries were written about all activities carried out in the three projects. Before starting the project we decided what should go into the diaries. The main effort at that time went into designing a check-list and thereby establishing a set of criteria.[7]

[6] (Naur 1972; Naur 1975; Naur 1983).
[7] (Nielsen 1984).

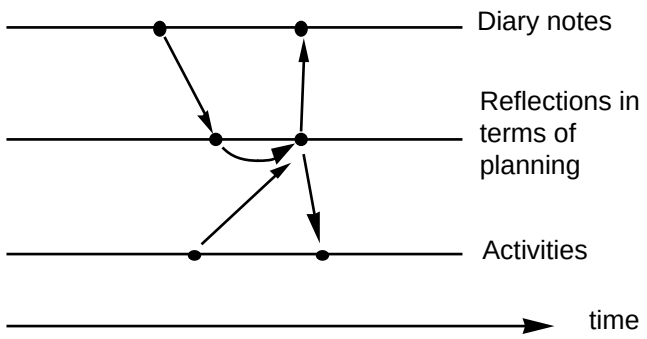


Figure 5.3: *Using diaries as a medium for planning activities. Planning is based on activities, previous reflections and diary notes and result in new diary notes and, hopefully, in new activities and new working practice. (Jepsen* et al. *1989, p. 212)*

The use of diaries is more than just a technique for documenting research, it is also a medium for management of projects in information systems development.[8] This means that while they are useful as documentation afterwards they are certainly also useful as a way of forcing the project members to think systematically about what they are doing when in the midst of it. Figures 5.2 and 5.3 illustrate two aspects of this.

5.2 Following A Single Methodology

The project in the North Branch is characteristic by its use of SA/SD as the one and only methodology. Such a faithful and straightforward use of the methodology represents one extreme in relying on a single methodology to be sufficient and operational.

Research Issues

The ideas guiding the research in this project are all covered by the overall issue: *What happens when a single methodology is used and it is used as literally as possible?*

SA/SD has been on the market for more than ten years and during this period a dozen books about how to practice it have been written. These classic textbooks together with the successful marketing of the

[8](Jepsen and Nielsen 1986; Jepsen *et al.* 1989).

basic ideas and benefits of the methodology have had a strong impact on how practitioners talk about information systems analysis and design. SA/SD is probably the most well-known in its field. It has this status despite the fact that it has been criticised from both theoretical and practical standpoints.[9] Nevertheless, the methodology is so significant that any work on methodologies will have to consider it carefully.

The methodology claims to cover the necessary aspects of information systems development not including the management aspects, i.e. the methodology covers the four activities of the model in Figure 2.1 on page 28. The significance of the methodology and its claim to cover all aspects are sufficient reasons for setting up an experiment where it is the one and only methodology.

The idea behind the experiment is to gain insight into the methodology in its present form by using it exactly as presented in the textbooks, as literally as possible. It must be impossible for others to claim that the lessons are based on malpractice or misconception of the ideas behind the methodology. The lessons learned must be based on a righteous use of the methodology. The issue then is: *What happens when* SA/SD*, its Weltanschauung, modelling language and frame of action, is used as prescribed?*

Even if the methodology was thought of and developed before participation was considered a major issue in information systems analysis and design DeMarco advocates and argues that data-flow models must be built together with the users. This raises the second research issue: *To what extent does* SA/SD *support participation?*

In the outset the methodology seems to fit the project. The systems developers know the methodology beforehand, the task is characterised by uncertainty, and a thorough analysis is necessary. From that viewpoint the methodology is a proper choice. But, what if the methodology during the project proves not to be sufficient or that some useful insight about banking procedures cannot be captured by the data-flow models? Such thoughts lead to a third research issue: *What are the limitations of* SA/SD *and when is it useful?*

The experiment can also be thought of in more general terms. SA/SD can fruitfully be seen as nothing more than a representative of the fully-covering and very detailed methodologies. From this viewpoint we can learn about the usefulness of using a single methodology as a stand-alone or as a 'universal' methodology. If we take the assumptions about methodologies from Chapter 1 seriously we should expect that there are

[9] Recently, some of the basic ideas have even questioned by Yourdon in (Coad and Yourdon 1990).

important limitations related to the belief that a single methodology can cope with all relevant aspects of the situation, i.e. there are unanticipated aspects of the situation that are necessary to cope with. Thus, the general research issue is: *What happens when a single methodology is used and it does not fit the situation fully?*

All these research issues conform with the themes for the whole research in the Royal Bank.

Course of Action in The North Branch

The project group consisted of two systems developers. They worked together analysing loan business in the North Branch. Basically, they had two main activities. Firstly, they interviewed bank clerks and validated data-flow models on meetings in the North Branch. There was approximately a one-day meeting each week for three months. Secondly, the rest of their efforts were put into fitting their analytical insight into the data-flow models. This took most of their time; from a single one-day meeting with the users they had sufficient information and confusion to work for almost a week afterwards. The models were checked and validated in cooperation with the bank clerks. Each model of the existing work with loans was presented and revised in sessions where the two systems developers and a clerk were present.

Initially, the systems developers knew nothing about, for example, building loans. To gain some preliminary understanding a context diagram was made of the work and the related data. A bank clerk working with building loans was interviewed by one of the systems developers. The other systems developer just listened to the interview and extracted the context diagram from this. During the interview the bank clerk would see what he said be transformed into a context diagram. If the systems developers or the bank clerk had questions related to the diagram they could clarify and correct the diagram immediately. This led to a first version of the context diagram for building loans which would be modified and revised by other bank clerks in similar sessions. All the data-flow modelling took place in very much the same way.

All the way through the project, the systems developers tried to do exactly what DeMarco advocated in his textbooks. This was not always possible, but they made an effort in trying to use SA/SD literally. By the end of the project, 110 pages of data-flow models had been made.[10] The 110 pages of models describe car loans and building loans in full detail at three levels: existing physical model, existing logical model and new

[10] (Jensen and Jepsen 1985a).

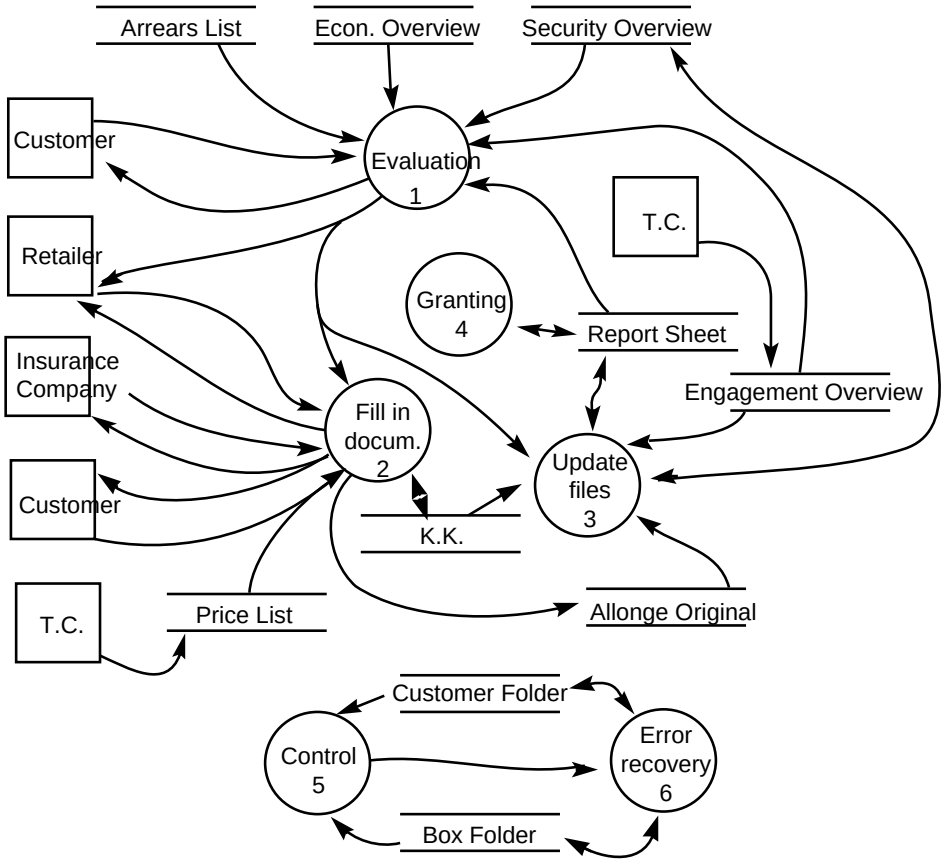


Figure 5.4: *One of the data-flow models made. The model describes a car loan at the existing physical level. The names of data dictionaries have been excluded in this figure. Key: →: data-flow; ◯: data process; =: file; □: sink or source of data; X: name of process or data.*

physical/logical model. Each of the models consists of context diagram, data-flow diagrams, transformation descriptions (M-specs) of the lower levels and data dictionaries. Figure 5.4 is an example of a data-flow diagram at a high level.

About SA/SD

Many lessons have been learned about SA/SD from this project. Only a small fraction of these are discussed here.[11] I find the following three lessons to be most significant and interesting with respect to the topic of this thesis.

Lesson 4 SA/SD *is based on the Weltanschauung that organisation and work can be seen as a computer system. Consequently,* SA/SD *is useful in domains characterised by routine activities. It is not suited for domains characterised by problem-solving.*

It is fairly easy to see from a theoretical viewpoint that SA/SD is based on a Weltanschauung where loan business is taken to be flow and transformation of data. By looking at working practice in the project in the North Branch we can see that SA/SD is also in a practical sense based on a computer Weltanschauung. From a practical view-point this is an important characteristic of SA/SD.

SA/SD provides data-flow modelling as a means of analysing work. Modelling work as data-flow is to view it as though it consists of data[12] and transformation of data. At any level in a data-flow model there is one or more processes that transform data input to data output. Data input comes from the environment or from another process. Data output is delivered to the environment or to another process, and some of the delivery of data may be mediated by a file. To model real world organisations and work implies in this case viewing it as though it were a computer. An information system is not only seen as a data system, but it is also a very simplistic data system where the basic metaphor is 'take data input and transform it into data output'. A data system in SA/SD is a batch-oriented system, and that makes it difficult to explicitly model features related to, for instance, on-line interaction.

[11] Other lessons and interpretations of the project can be found in (Jensen and Jepsen 1985b) and (Bengtson *et al.* 1985a). The citations in this section stems from (Jensen and Jepsen 1984).

[12] Note that 'data' according to The IFIP Dictionary is: a representation of facts or ideas in a *formalised* manner capable of being communicated or manipulated by some process. (A1). (Gould 1971).

A loan business in The Royal Bank, whether it is a private loan, building loan, re-mortgaging, or student loan, is always handled by the same bank clerk all the way from start until end. Any loan business is in principle handled as a special case. There is variation with respect to what information is needed to handle the case and how it can be obtained. This means, for example, that there is no 'typical' customer. Surprisingly, the project group met almost insuperable obstacles in modelling the case-based nature of the work. The following was recorded in the diary:

> Some information one [a bank clerk] can always remember, some is retrieved if in doubt, and some is always retrieved. It is a continuous spectrum.
>
> That which is always remembered, is specified in mini-specs. That which is retrieved, is shown as a file. There is a choice [imposed by the methodology] between these two extremes.

SA/SD was frustrating for the systems developers. The inherent variation in loan business could not be captured in a data-flow model. The systems developers had some insight into this variation, but SA/SD did not support analysis of this variation. The implication is that SA/SD is not useful when the domain is characterised by case-based work, not to speak of problem-solving in general.

Being based on a computer Weltanschauung SA/SD is, however, useful when the domain is characterised by routines. If the work in the domain is routine then it is possible and often useful to use SA/SD to analyse it. The procedural nature of routine activities can be captured as processes of data transformation. The reason for modelling procedures in this way is that such processes are easy to implement in a computer since they already have an algorithmic and formal outlook. Therefore, SA/SD is a methodology for automation of routine work.

The modelling language is the most important characteristic of SA/SD. It can be claimed that SA/SD is nothing more than a modelling language. This is not entirely true, but the Weltanschauung is to a large extent determined by the modelling language.

Lesson 5 SA/SD *supports the development of a coherent set of reductionistic models of the information system viewed as a data processing system. The models can give an overview, but they do not provide insight into the information system as a whole.*

The models built by means of SA/SD give a useful overview of loan business. In the diary we find:

> By that time the DFD took up three sheets of A4 paper and contained 16 processes (DeMarco thinks that about 7 at each level is maximum). Gradually we lost the overview. We solved the problem by "going up a level" in the DFDs. A DFD with only six processes came out, and that gave us an overview of building loans.
>
> We had started out in the middle of the hierarchy — we knew this, [and] additionally we knew why we had done it. We started out in the middle because we wanted to get information about building loan and the associated banking. The "new" DFD was already on the board 14 days ago, but at that time it did not tell us anything for the simple reason that we did not know what banking was about (what was hidden inside the processes). It gave us no information about banking, but together with our new insight it turned out to be a great help in gaining an overview.

This extract illustrates in what way SA/SD provided the systems developers with an overview of loan business. To some extent it also illustrates the coherence between the models. Coherence is basically a relationship between levels, in the sense that a higher level is an abstraction of a lower level and vice versa. But coherence also means that even though SA/SD supports the building of 'different' models these models describe a part of the real world from the same Weltanschauung, i.e. based on the same assumptions about the real world. The logical model of the existing loan business described the real world in exactly the same manner as it is described in the physical model of the existing. The models were coherent in the sense that they formed a single description of the real world 'as it is.'

There is nothing at a given level in a data-flow model that cannot be explained by looking at a lower level. There are no emergent properties at a given level. All input and output can be seen at a lower level and a process is a simple way of clustering lower-level processes. A data-flow model is logically a set of atomic processes and the flow between them and all higher-level models only provides an overview where some details are left out. In this sense SA/SD supports the building of reductionistic models.

Lesson 6 SA/SD *sees systems developers as model builders and users as model checkers. The models can be understood by the users, but it requires analytical skills to build a model.*

It is fairly easy for the users to understand data-flow models.

> The papers [data-flow models] we used to form the basis of the interview were excellent as tools of control. We did indeed succeed in keeping the conversation on the issues we wanted and in guiding it towards new issues [typically processes further down the hierarchy].

The users understand the models basically because of the hierarchical structure. At any time they will only have to look at a given level of the models (7 ± 2 processes) and can for a moment forget about the rest. If it is necessary to look at it in more detail then it is only a matter of 'going down a level'. The models are also easy to understand because the basic concepts are very simple and complexity does not increase when they are combined. The systems developers experienced that the users could take part in a constructive dialogue where the models were modified if mistakes were found. In most cases, however, the users would function as model checkers answering the question: Is the model correct now?

The users cannot, however, necessarily build the models themselves. It only takes insight into the concrete work to check the correctness of a model. But it takes analytical skills to build the models because they express abstractions of the actual concrete contents of the work.

It is not the modelling language that determines whether the methodology can be used to undertake participatory analysis and design. The modelling language can facilitate or inhibit participatory activities, but it is insufficient as a support for such activities. To achieve participatory analysis and design a methodology must have a strong participatory frame of action.

About Following a Single Methodology

While the above lessons were specific to SA/SD the following lessons are more general about the working practice based on a single methodology.

Lesson 7 *The idea of using only a single methodology is not feasible. In a specific situation there will always be relevant matters that the methodology does not address.*

Lesson 4 showed that SA/SD has limitations. It is useful in domains characterised by procedures and routines, but it is not useful in domains characterised by problem-solving (or simply case-based work). There are relevant aspects of loan business that cannot be understood by means of SA/SD. Of course, the systems developers experience this and get frustrated about this limitation; but the case-based aspects are outside the domain of application of SA/SD. By using SA/SD as the one and

only methodology, the systems developers are unable to analyse the case-based aspects of the work in a structured and systematic way.

All methodologies have limitations. The limitations may very well be different from those of SA/SD; but they are there. Using only one methodology in a project creates difficulty for the systems developers. The difficulty arises when relevant matters of the situation at hand escape the modelling language. The systems developers may be aware of these difficulties, but the methodology does not provide them with a Weltanschauung or modelling language by which they can deal with these matters in a satisfying manner. Thus the idea of using only a single methodology is not feasible.

This lesson does not necessarily apply to methodologies where the Weltanschauung with respect to the domain is rather weak, i.e., without a theory about the domain or without a domain-specific modelling language. The point is, however, that methodologies that have been developed for information systems analysis and design are based on Weltanschauungen that make assumptions about information systems one way or the other. This means that they will incorporate an implicit or explicit theory about information systems or a modelling language specific to information systems.

Not only is it difficult to use only a single methodology; it is also a potential risk.

Lesson 8 *By using a methodology systems developers adopt a specific Weltanschauung. They tend to fit the real world into this Weltanschauung rather than being curious and critical. Furthermore, they impose this Weltanschauung on the communication with the users.*

The following extract from the diary illustrates the blindness and the lack of curiosity involved in using SA/SD.

> 1. It was discovered in a review of ideas for design concerning a specific type of loan that there are two different ways of handling this type of loan. The group had not realised this earlier even though the group had been through the description with three different bank clerks. What separates the two ways is who (car dealer or bank) fills in the documents. That is determined by the availability of another document.
> 2. After several conversations with bank clerks the group perceived that building loans consist of two loans: one for building the house and a new loan to take over the first loan when the house is finished. By coincidence the group read

> an advertisement about building loans and the group realised that the building loan only exists during the building of the house and is taken over by a mortgage loan.

The misunderstanding could in both cases have been avoided if the systems developers had asked: Why? But SA/SD does not support that. The primary reason for this is that it is outside the scope of the Weltanschauung and the modelling language to address the above issues. In other words, the systems developers have adopted the built-in Weltanschauung to such an extent that they are not any longer investigating issues that are difficult to give meaning to by means of the methodology's Weltanschauung.

A methodology imposes its built-in Weltanschauung on the systems developers. It also imposes its Weltanschauung on the communication with the users. Lesson 6 showed that though the users can understand the models they are less able to build the models themselves. The systems developers have the task of building the models; and this can only be done by communicating with the users about their work. The agenda for the communication between the systems developers and the users is then set by the methodology. Thereby the Weltanschauung is imposed on the communication.

5.3 Theoretical Support

Instead of striving to use a fully operational methodology the opposite standpoint was taken in the project in the East Branch, namely that a general theory of organisations and information systems could form the basis of analysis and design.

Research Issues

Transaction Cost Theory (TC) is an organisation theory where the focus is on contracts that regulate transactions or exchange between partakers.[13] An organisation is viewed as the institutional, contractual arrangements that govern a network of transactions. Information is used when defining and implementing these contracts, or more precisely:

> The information system of an organisation is then transactionally defined as the network of information flows that are needed to cre-

[13]The theory has been developed for information systems in organisations by Ciborra (1981) based on the general organisation theories of Ouchi (1980) and Williamson (1979).

> ate, set up, control, and maintain the organisation's constituent contracts.[14]

The aim is to reduce the cost of handling contracts by providing useful information.

TC is not at the outset operational, i.e. it does not provide many practical guidelines. It is evident that the project must to a large extent rely on a combination of theory and the systems developers general competence in computer science and information systems analysis and design. The issues investigated in this project can be summarised into: *What happens when a general theory is used to support practical competence in information systems analysis and design?*

The theory is abstract and is about the structure and transformation of organisations. It does not contain any knowledge about the process of understanding an organisation by means of the theory, i.e. it is both in principle and in practice ontological. It needs to be made much more operational in order to serve as guideline for analysing and designing information systems. In the case of SA/SD we were interested in the use of it, but in this case we are interested in both the making of something potentially useful based on TC and its use. Thus, the specific research issue regarding TC is: *How can* TC *be made operational and used in practical analysis and design?*

Since it is only a theory about organisations it is not at all obvious that it is usable in a participatory context. It is not hard to envisage the inherent difficulty in communicating with users in terms of transactions, contracts, and transaction cost. A research issue then is: *What are the possibilities for participation when using* TC*?*

Even though TC is not in any sense described in detail in the literature we will also have to consider the issue: *What are the limitations of* TC *and when is it useful?*

It is interesting to see, at the general level, how a theory may be used in information systems analysis and design. The usual strategy for bringing past experience and knowledge into a new project is to use a methodology with some operational strength. However, it may be equally useful to rely on a general theory and on the systems developers to adapt and make the theory concrete in a particular situation. Therefore, a general research issue is: *What is the relationship between practical competence and a theory which is made operational?*

[14] (Ciborra 1981, p. 309).

Course of Action in The East Branch

According to TC, contracts are negotiated and regulated through a series of processes, called the transactional life-cycle, consisting of: searching partakers, establishing contracts, controlling and regulating contracts, and maintaining contracts. The transaction costs are then measured as the costs of the transactional life-cycle, i.e. the resources used for performing the search, establishment, etc.

According to Ciborra, the transaction costs can be reduced by an improved use of information technology. Information technology may reduce the costs by extension of linkages between the partakers in the exchange governed by the contracts in question. It may also reduce the costs by standardising the transactional life-cycle.

The theory is not operational, but it provides a Weltanschauung and parts of a language for understanding the use of information systems with the purpose of reducing transaction costs. During the project an operational form of the theory was sought where it was possible by analysing the existing organisation and its information to get ideas about new or improved use of information systems.

The project group decided to work in depth with the transaction between the customers and the branch. Loan business is characterised by a network of contracts of a complexity much higher than it is possible to investigate by means of the theory. Despite this, a focus was chosen where it was both interesting and possible to do an in-depth analysis. Furthermore, the group decided to measure transaction costs by the frequency and weight of the necessary information. Frequency is a measure of how often some information is needed. Weight is a measure of the complexity of the information.

TC provides the fundamental concepts, but the details of how to model a specific organisation by means of these concepts were developed by the project group. For each type of loan, four models of transactions and contracts were made:

- An overview model focusing on which activities there are and their sequence.

- A partaker model identifying partakers and contracts and focusing on changes in the partaker relationship. An example is shown in Figure 5.5.

- An activity model where each of the activities in the overview is described in detail with respect to the four transactional activities and their placement in the partaker model.

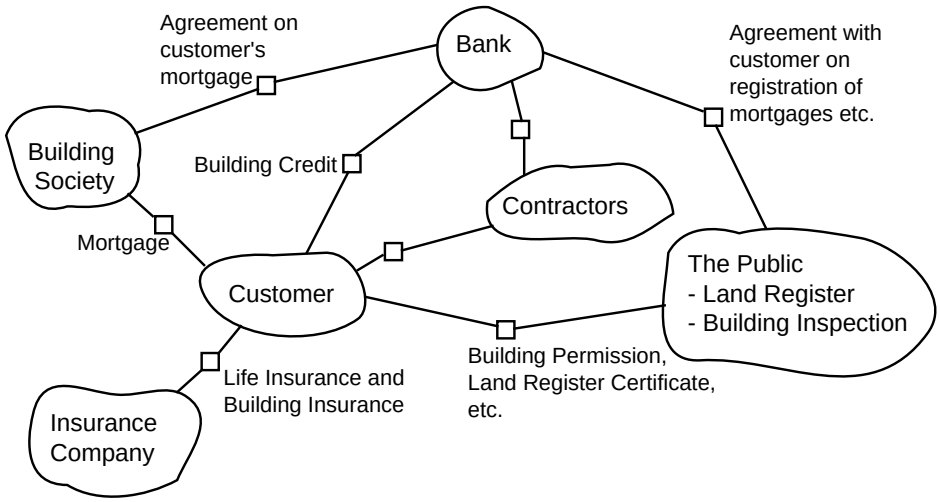


Figure 5.5: *One of the partaker models used. This model describes the partakers in a building loan seen from the standpoint of the customer. Key: ◯: partaker; □: contract.*

- A transaction cost model that identifies the information needed and produced by each activity in the activity model together with an evaluation of frequency and weight of this information. An example is shown in Figure 5.6.

The transaction cost model formed the basis for an assessment of a possible reduction of transaction costs by introducing new or improved information systems.

The various models were made on the basis of interviews with selected bank clerks in the East Branch. It was not an easy task for the bank clerks to understand the descriptions and a convincing form of presentation was never reached.

About TC

In many ways it was difficult for the East Project to obtain a suitable working practice. They were always in the middle of trying to change it. Having set out to use a theory instead of a methodology they had to use considerable time on adjusting and adapting the theory to the particular needs of the situation. Many lessons can be learned from this,

-
-

1. SEARCH PARTAKERS

The bank clerk must decide whether to commence negotiations or not.
— Frequency: Often.
— Weight: Easy, if the customer is already a customer in the bank.
Difficult, if the customer is not already a customer or never has been:
talk to the customer, information from Land Register, etc.
— Needed information: Engagement Overview Document, Report Sheet, Account Overview.
— Evaluation: Commence negotiation (Yes/No)?

2. ESTABLISH CONTRACTS

Far the most voluminous part of the work.

2.1. Building Loan Form
A huge task that often takes several meetings with the customer. It involves many difficult activities: information search, information retrieval, computations, evaluations, decision taking, and consulting.

-
-
-

Figure 5.6: *Part of one of the transaction cost models used. This part describes the search for partakers' activity in the transactional life-cycle of a building loan.*

and I have chosen to focus on what I find to be the most significant and relevant lessons.[15]

Lesson 9 TC *is based on the Weltanschauung that organisation and work can be seen as transactions governed by contracts. It is a useful framework for understanding information needs of administrative organisations.*

TC is based on a Weltanschauung where any activity is seen as a transaction governed by a contract. In order to analyse the organisation we must identify and understand these transactions and contracts reducing the costs of the transactional life-cycle by improving the information systems.

It was the project group's experience that they did, in the end, arrive at some useful insight about loan business. There may be at least two reasons for this which are not mutually exclusive. On the one hand, they achieved this because they were able to combine the theoretical concepts and ideas of TC with their own intuition and practical competence. They ended up having learned more about TC than about loan business, but the process they went through provided them with a number of simple ways of applying the general theory, i.e. the theory became operational to some extent. On the other hand, it is fair to state, based on the experience from this project, that TC in itself is useful in understanding domains consisting primarily of administrative activities like banking. It is characteristic for administrative activities that the information systems they use provide information *about* other activities. Therefore, if we want to improve the information systems related to an administrative activity it may be useful to look at how we can reduce the costs of administering the contracts.

It is interesting to notice that it does not matter to TC whether the domain is characterised by procedural tasks or problem solving tasks as it did with SA/SD.

The modelling language of TC is one of its weak points. However, TC does encourage modelling the domain from several standpoints.

Lesson 10 TC *invites the systems developers to build several models expressing different levels of detail and different contracts.*

[15]Other interpretations of the working practice in The East Project can be found in (Munkholm and Steffensen 1985; Munkholm and Steffensen 1986) and (Bengtson *et al.* 1985a). The citations in this section stem from (Munkholm and Steffensen 1984).

TC does not in itself contain many ideas as to how to model an organisation and its information needs. Coming from a tradition of primarily computer science ideas, the project group believed in the beginning that they had to come up with *one* model of loan business. They soon realised, though, that it was not possible to model activities and contracts related to loan business in such a straightforward manner. The contents and relevance of the contracts they were able to identify depended on the standpoint they took. For example, the relevance of contracts related to a building loan depended on whether they took the starting point in the customer, the bank, or the building society.

It is fair to say that TC invited them—and sometimes forced them—to build several models. The four types of models already mentioned express different levels of detail ranging from the overall model, via the partaker model and the activity model, to the transaction cost model. Furthermore, within the partaker model different contracts appear that basically govern the same transaction, but they express the governing arrangements from different viewpoints. For example, from one standpoint the customer and the bank are distinct partakers with an intermediate contract, and from another standpoint the bank and the customer form a partaker together and relate to a building society.

TC is useful in producing a rich understanding of administrative information systems, but it is, at the same time, very difficult to use.

Lesson 11 TC *is still insufficient as a frame of action. The concepts of* TC *are too imprecise and abstract, and the users do not understand the models sufficiently.*

The concepts that TC offers are much too imprecise and abstract. The project group experienced, for example, that the theory gives the impression that contracts (that govern transactions) are fairly stable; but real world organisations are different since contracts are changing all the time with respect to their importance, contents, information needs, and costs.

The diary reveals another difficulty as the four transactional activities can only in principle be fully separated, but real world activities that contribute to the transaction costs cannot in practice be fully separated when depicting them in a model. Furthermore, TC offers no guidance in defining operational measures of transaction cost and yet 'transaction cost' is the most fundamental concept of TC. The project group did make some of the theory operational, but the concepts remained imprecise, though less abstract.

There is only an imprecise modelling language available and the users could not understand the models sufficiently. The following extract from the diary illustrates this:

> We go through the model from one end and they explain the mistakes and misunderstandings we have made. ...
>
> They haven't assessed the activities in terms of frequency and weight as we had asked them to do. We perceive that they find it difficult to assess their work on information processing based on the categories.

It is difficult to undertake participatory analysis and design based on TC. The project group tried in a nearly extreme way to create a true participatory setting where it should have been possible to interact with the users in analysing the existing loan business and designing new information systems. The success of such efforts seems to depend on whether the modelling language they tried to establish was meaningful to the users. A modelling language based on TC does not make sense to a user. Not even analytically minded users find the fairly simple measures of frequency and weight meaningful.

About Using a Theory

The above three lessons are specific to TC. Some of these lessons may be generalised into the following lesson about using a theory in general as support for practical competence.

Lesson 12 *A theory provides a Weltanschauung, parts of a domain-specific modelling language and an insufficient frame of action. Effective use of a theory requires not only analytical skills but also considerable resources for developing a working practice based on the theory.*

A theory may be useful in several ways, e.g. as a set of concepts together with some understanding of information systems that makes these concepts meaningful, or as a set of concepts which can be used to model aspects relevant to information systems. As with TC it is fair to claim that a theory will at least provide a Weltanschauung relevant to information systems. The Weltanschauung is implicitly reflected in the concepts provided by the theory. These concepts can be seen as an informal modelling language which will be fairly abstract and therefore only a part of a necessary (or full) modelling language. It is, however, domain-specific in the sense that the domain is concerned with information systems, e.g.

in the case of TC the modelling language is relevant for modelling organisations from the viewpoint of contracts, cf. Lesson 9. However, a theory has only a weak frame of action. Basically, this is what distinguishes a theory from a methodology. The frame of action is weak partly because the concepts of the theory (i.e. the abstract modelling language) are not immediately applicable. See Lesson 10 for the specific argument about TC.

All this requires that the systems developers have the necessary analytical skills for using the theory despite the fact that it is not immediately applicable. The analytical skills are necessary for carving up the phenomenon in question and finding relevant boundaries for the concrete appearances of the concepts of the theory. The analytical process is interactive between moving the understanding of the phenomenon in the direction of the concepts provided and moving the concepts in the direction of the phenomenon. But the general analytical skills of the systems developers are not sufficient. To develop a working practice from a theory also requires that resources are used on considering modification and adaption of the theory. The modification and adaption will have to be done explicitly (unlike some methodologies) because the modelling and the use of the models need to be operational and to produce a usable outcome. TC, for example, provides the concept of transaction cost but no operational measure is defined and the project group had to find the measures 'weight' and 'frequency' by themselves.

5.4 Combining Methodologies

The two projects described above used methodologies based on a computer Weltanschauung and on an organisational Weltanschauung, respectively. The third project was set up to bridge the gap between two such Weltanschauungen by using a combination of methodologies.

Research Issues

The research work in the South Project was to a large extent guided by the embodying issue: *What happens when a technical methodology is combined with models of office work?*

JSD gives the impression of being an alternative to methodologies like SA/SD.[16] A major difference pointed out by Jackson in the introduction to the original textbook is that the functionality of the information system is considered more unstable than the subject area of the system.

[16] (Jackson 1983, p. 12).

Hence, first a model of the subject area should be made as a stable framework on which the more volatile functions can then be built. In SA/SD the functionality gets primacy attention. For this reason it is interesting to make an experimental comparison of SA/SD and JSD. Hence, a research issue is: *What happens when* JSD *is used for handling technical aspects?*

Even though JSD is developed at a late stage, at a time when participation as an ideal had been around for some time, it is not at all obvious that it fits neatly with these ideas. We know in advance that it focuses attention on the technical aspects indicating the non-participatory character, but on the other hand, JSD is a methodology for designing information systems thus in principle it should be done participatorily. Hence, an issue to be investigated is: *To what extent does* JSD *support participation?*

Some of the limitations of JSD are known beforehand, but what if they show to be more different than anticipated? We will still have to pursue insight into the issue: *What are the limitations and the areas of usefulness of* JSD*?*

Like DeMarco, Jackson claims that JSD covers all the activities of information systems analysis and design except the management of it. It is evident, however, that JSD is more tightly connected to the computer perspective as it is impossible to model the use of the system or even simple aspects of the context of work where the system is to be used. With this in mind, the project group decided to supplement and combine JSD with Newman's approach for modelling office work (MOW).

Newman presents a framework for understanding office work where the work in an office consists of office procedures and office problem solving, which is used when the procedures fail.[17] The framework should in principle be useful in analysing the organisation. The research issue is: *What happens when* MOW *is used for handling organisational aspects?*

As with all the other approaches we are interested in the scope of it together with the special interest in participation: *To what extent does* MOW *support participation?* And: *What are the limitations and the areas of usefulness of* MOW*?*

The above research issues relate to the specific approaches, but as with the other projects it is possible to raise more general issues. The research issue of multi-viewing put forward in Chapter 1 can be investigated by means of this experiment. Thus, an important general research issue is: *To what extent and how can methodologies based on different Weltanschauungen be combined?*

[17](Newman 1983).

Course of Action in The South Branch

There are three types of interconnected models of office work: process models, task models, and procedure models for the description of future and present offices.[18] Newman bases his office models on the observation that office work consists of office procedures and problem solving. The procedures are established ways of handling a case. They are both the official explicit procedures and the personal more or less implicit procedures that have been established through working in the office. The procedures are used when handling a case which may range from the often-experienced case handled as routine to the seldom-experienced case handled more carefully and thoroughly. Office workers will from time to time be confronted with cases which none of the existing procedures can deal with. Such cases will have to be handled as problem solving. In problem solving the office workers must find a new way of dealing with the case. Existing procedures are of importance in problem solving because office problems may be solved by using parts of existing procedures or by combining procedures.

Process models focus on the top level at office functions of which there are four categories in any office: primary functions, organising functions, contract related functions, and support functions. The intermediate level relates to the office processes that depict the activities through which the functions are performed. The third and lowest level is devoted to office methods that are alternative and different ways by means of which the office processes are performed. An example of how the project group chose to model a process is shown in Figure 5.7.[19]

The task models focus on office tasks. An office task is a unit of work office workers use in organising and talking about their own work. Figure 5.8 gives an example of such a task as part of a model. The project group chose to characterise office tasks by their origin and definition. The definition is composed of: purpose, tools, qualifications, relation to other tasks, and allocation.

[18] (Newman 1983) was our main reference. The reference is two chapters of a draft for (Newman 1986). When we used the ideas from the draft in the Autumn of 1984 we did not know that only the chapter on the general outlook of offices would be in the final book and that the chapter on models of office work would be finally excluded. We learned this when we presented our use of the models to Newman in 1985; we also learned that Newman never intended to actually build the three types of models; to him they are merely ways of thinking about offices.

[19] (Bengtson *et al.* 1984a) contains the project group's interpretations of (Newman 1983) together with some ideas as to how to actually build the models outlined by Newman.

•
•

FORMAL GRANTING OF PRIVATE LOAN

* registration of security	* type-in securities in SRS, sometimes based on Help From SRS * fill in Security Sheet
* registration of loan based on loan information	* information is written on Report Sheet in the standard format (Account No from Account No List): +---------- | Date | Account No Loan Kr. YY | <description of purpose> ... | Engagement X X X X | New Loan X X | After Granting X X X X +----------
* granting	* own granting, Branch Manager signs on Report Sheet * Branch Manager grants on Report Sheet * Branch Manager recommends, Credit Head Office grants on Report Sheet (internal mail)
* refusal	* send letter * telephone conversation * conversation in the bank

•
•
•

Figure 5.7: *Part of one of the process models used. This part describes the process where a private (non-business) loan is formally granted. Key: the left-hand names denote some actions to be taken, e.g. some loans are 'granted' some are 'refused'; the right-hand side is different ways of carrying out the actions, e.g. small loans can be granted by any bank clerk, medium-sized loans by the Branch Manager and large loans by the Head Office.*

-
-

-------OO-------

REGISTRATION OF PRIVATE LOAN (<== FORMAL GRANTING)

purpose	: registrate the loan so the granting can be formally established
tools	: "Report Sheet", Engagement Overview, Rule of Credit, typewriter, account list
nec. qual.	: knowledge about own granting right and about how to write "Report Sheet"
relation	: prerequisite that EVALUATION OF PRIVATE LOAN is carried out
allocation	: all bank clerks can do the task; typically, the task will be allocated when the bank clerk did EVALUATION OF PRIVATE LOAN her/himself, or the task is transfered from another clerk

------OO------

-
-
-

Figure 5.8: *Part of one of the task models used. This part describes the task related to the office process FORMAL GRANTING. The task is named REGISTRATION OF PRIVATE LOAN and has five characterising aspects. Key: $\Longleftarrow$: the process that this task supports.*

The procedure model is a very detailed model of the range of procedures that are typically used when performing the office tasks.

Because JSD is primarily concerned with design it was decided to start the analysis by understanding the present work with loans in the South Branch by means of the office models. After having modelled the office work JSD could then be brought into use for designing a computer system to support the work. Afterwards, the future banking was described in a modified set of office models.

The office models of the present work with loans in the South Branch were made partly by the project group of three systems developers based on interviews with a number of bank clerks. When the initial impressions from the interview had been described in the models they were revised and modified in sessions where the systems developers and the bank clerks, mostly one at a time, were present. At the beginning some time was spent explaining the models to the clerks to enable an understanding and later a constructive criticism of the models. In the end, the bank clerks, according to their own statements, had gained interesting insight into their own work, and the systems developers had, by being in charge of making the models, gained an understanding into the work with loans that enabled them to move on to design of computer-support of that work.

With a few minor alterations of JSD a computer system was designed and described. The alterations of the methodology had mainly to do with the use of informal models where JSD demanded formal models. The informal models were used in sessions with the bank clerks, for example, to establish the functionality of the computer system, because it was felt that the formal models of JSD would not be comprehended by the clerks. When the functionality had been established the informal models were translated into the required formal models by the project group without discussing them with the clerks.

After having designed the computer system attention returned to the loan business where it was going to be used. This was done by taking all the office models of the present loan business and systematically changing all details so that they would match the use of the new system. An example of such changes are shown in Figure 5.10. The changes were made in hand-writing in order to be visible and the actual changes of the work could be locally determined and discussed. All the models of the future work were then discussed in sessions with the bank clerks in order to establish the benefits and drawbacks of the changes in their work.

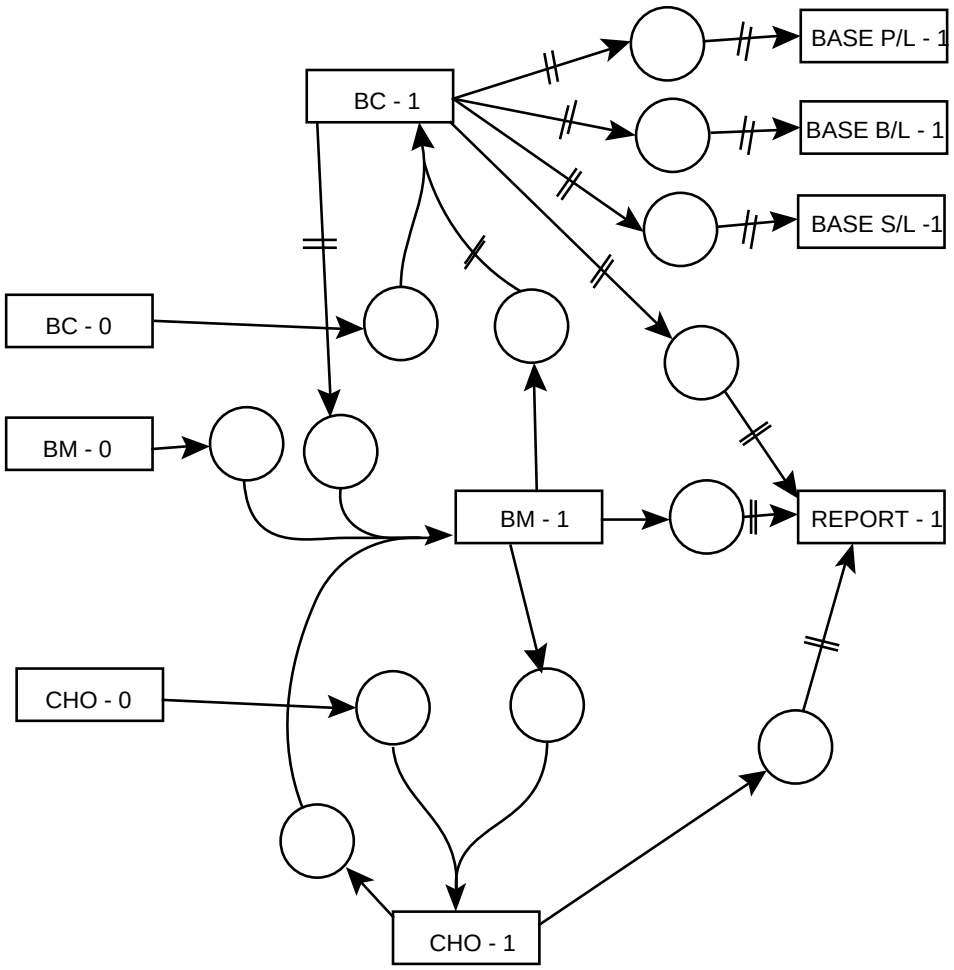


Figure 5.9: *The initial model used. Key: □: entity; ◯: data stream connection; ◇: state vector connection; BC: the entity BANK CLERK, BM: BRANCH MANAGER, CHO: CREDIT HEAD OFFICE; -0 : real world entity; -1: initial model entity.*

•
•

FORMAL GRANTING OF PRIVATE LOAN

* registration of security		* type-in securities in SRS, sometimes based on Help From SRS ~~* fill in Security Sheet~~
* registration of loan based on loan information		*filled in* *PC-Report* * information ~~is written~~ on ~~Report Sheet~~ in the standard format (Account No from Account No List): +---------- | Date | Account No Loan Kr. YY | <description of purpose> ... | Engagement X X X X | New Loan X X | After Granting X X X X +----------
* granting	** PC-Transfer from PC-Base SRS, OIS; the rest is filled in and edited*	* own granting, Branch Manager signs ~~on Report Sheet~~ * Branch Manager grants on ~~Report Sheet~~ * Branch Manager recommends, Credit Head Office grants on ~~Report Sheet~~ ~~(internal mail)~~ *+ PC-Mail*
* refusal		* send letter * telephone conversation * conversation in the bank

•
•
•

Figure 5.10: *Part of one of the modified process models used. This part describes the process where a private (non-business) loan is formally granted. The modification was made in hand-writing in the original to make it visible. Key: the left-hand names denote some actions to be taken, e.g. some loans are 'granted' some are 'refused'; the right-hand side is different ways of carrying out the actions, e.g. small loans can be granted by any bank clerk, medium-sized loans by the Branch Manager and large loans by the Head Office.*

About JSD and MOW

We learned in this project, as in the other two projects, many lessons that are specific to to JSD and MOW.[20] A number of these contribute to the inquiry undertaken in this thesis.

Lesson 13 JSD *is based on the Weltanschauung that organisation and work can be seen as a computer system.* JSD *is useful in domains characterised by procedures and problem solving. But it requires that the domain is characterised by time-ordered events and absence of existing computer-based information systems.*

In the South Project it was not surprising to learn that JSD is indeed based on a technical Weltanschauung where loan business is viewed as though it is a computer system. The Weltanschauung was exercised by building a model of the real world, the Initial Model, and functions that provide data about the real world. The Initial Model contains the entities in the real world which the computer systems need to know about. The real world is modelled by naming these entities and for each of them the actions they can perform. Relationships between the entities are modelled as transfer of data. This may seem to be equivalent to SA/SD, but the main difference is that by using SA/SD we look for processes that transform data while using JSD we look for entities that contain data and change their state by receiving data from other entities.

For example, the model built in this project, see Figure 5.9, contains the bank clerks and other users as entities.[21] The actions that the entity BANK CLERK can perform are all actions related to loan business, e.g. GRANT LOAN. The Initial Model simulates the real world. If a bank clerk grants a loan then the corresponding BANK CLERK will perform the action GRANT LOAN. The action GRANT LOAN is initiated when BANK CLERK receives data on the data-stream connection with the real-world bank clerk. Thus, by using JSD the systems developers view the work of the bank clerks as being expressible as a computer system.

[20] Some other lessons and interpretations can be found in (Bengtson *et al.* 1985b) and (Bengtson *et al.* 1985a). The citations in this section stem from (Bengtson *et al.* 1984b).

[21] This can be called a mis-interpretation or a re-interpretation of Jackson's ideas depending on the willingness to observe that methodologies are always adapted to the specific situation when used. Jackson (1983) takes a bank as an example and builds a model with the customer and not the bank clerk as an entity. He does this by arguing that we need to know about the customer's actions with respect to an account and not about the clerk's actions with respect to the customer's account. In the South Project it was decided to let the bank clerk become an entity because according to the remit we should design information provision for the clerks not automate the clerks' work.

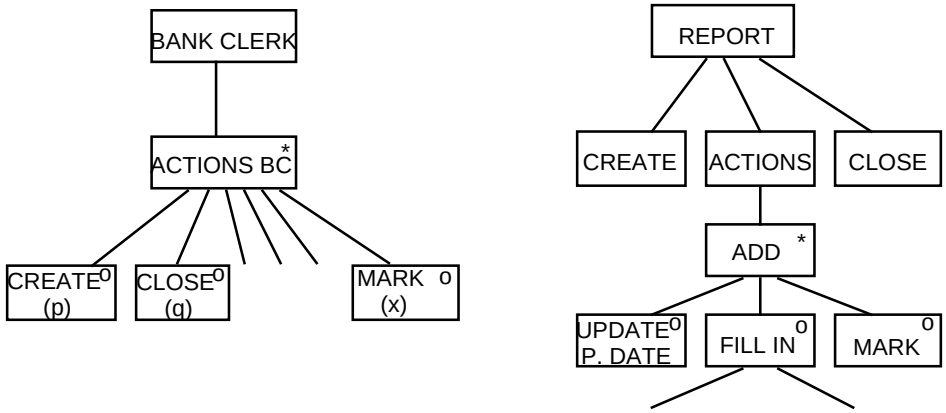


Figure 5.11: *Two entity specifications. BANK CLERK without ordered actions. REPORT with ordered actions. Key: ∘: choose one of the sub-actions; ∗: iteration of sub-actions from left to right; □: action.*

The project group found JSD useful with respect to both procedural and problem solving aspects of the domain. Procedures are easily captured because they can be modelled as a sequence of actions. Problem solving aspects can be modelled by pointing out which actions they involve but without prescribing a sequence in which they should be performed. Loan business in The Royal Bank is to some extent characterised by problem solving because any loan is basically unique and a bank clerk can in principle work on many loans at a time. This means that the bank clerk can be modelled as performing a set of actions without any restriction as to the order in which they should be performed. Figure 5.11 illustrates how BANK CLERK is without ordering. It also illustrates that the ordering of actions with respect to a specific loan is brought about by REPORT that has common actions with BANK CLERK. In this way it is fairly easy to model both procedural and problem solving aspects without even noticing the distinction when modelling.

So far, it seems as if the Weltanschauung of JSD makes both procedures and problem solving meaningful. But there are limitations to this. In the case of loan business the structure of a loan brings ordering of actions into the models. In general the domain must be characterised by time-ordered events or actions one way or the other. If there is no time-order then all entities will have unordered actions, like BANK CLERK, and the Initial Model will thus be uninteresting. It will just be a list of actions.

Procedures and problem solving did not create any particular obstacles for the project group. But the fact that the new computer-based information system should operate in connection to and simultaneously with at least two other computer-based information systems did create almost insuperable obstacles. It is impossible, in the Initial Model, to model such a relationship and it was therefore necessary to invent a notation for modelling it anyway. In the model it is shown as a state vector connection when an entity receives data from another information system and as delivery of output when another system takes data. This means that the very idea behind JSD is that a total computer system is being designed and not only parts that relate to other systems.

It is important to understand the Weltanschauung of JSD; but it is equally important to understand the nature of the modelling language and for what purposes it may be used.

Lesson 14 JSD *supports modelling of the details of entities and functions together with their overall relationship. The models are useful for implementation considerations; but they are difficult to understand by the users.*

The models made by means of JSD have two levels. At the top-level we see all the entities, functions and their relationships in terms of data stream and state vector connections. Each entity and function is modelled in terms of the sequence (or ordering) in which the actions of it are to be performed. Since the actions are seen as atomic-like events without time-span, and since all necessary data attributes are added, the entities and functions are specified in full detail. Going up a level from the details of actions, the computer system is specified at a more overall level as entities (functions are also modelled as entities) and their relationship. The model is thus giving an overview of the computer system without explicitly treating it as a whole.

Such a model is useful when considering how to implement it on a computer. The entities and their connections are specified in a language very close to a programming language. If the programming language is object-oriented like the modelling language of JSD it is fairly easy to implement the model. If it is a traditional programming language, the model needs to be translated, but in principle the translation is not difficult. The case that the information systems in the end are going to be computer-based makes JSD a useful methodology.

On the other hand, the models are not at all easy to use when communicating with the users. The following extract from the diary illustrates this:

2. First, we presented the entity BANK CLERK: they had difficulties in grasping the idea that all actions are unordered (i.e. without time order), but the actions are alright.
3. We got better words for some of the actions.
   ...
5. Next entity REPORT: The actions are time ordered and we explained the symbols in the diagrams at the same time. They understood this description much better than the other entities.
6. They said that it was easier to understand. That is one of the reasons why they were very constructive and we got many new words for actions and the entities ordered in another way.

In communicating with the users, as in the above extract, the project group used the early stages of the Initial Model, i.e. lists of entities and actions together with the structure of the entities. The more complex model containing relationships and functions were never used in sessions with users. The project group therefore decided to discuss functions with the users based on, what they called, an informal function model.

2. Since we are working in the bank on Wednesday it is important that we have something to present. We agree that Initial Model Step and Function Step are *not* going to be presented in Jackson's form.
3. We chose to present proposals for functions in prose descriptions. For each idea (report sheet, computation of private, car, student, and building loan) we described possible functions and attributes.

This was felt necessary by the project group since they had no expectation that the users could under any circumstances understand the more technical models. The informal function model worked well, but it was the project group's experience that it could not have been more formal than that.

Newman's framework for understanding offices, which was taken as though it was a methodology, was remarkably easy to use. It required some interpretation of his ideas, but afterwards it was reasonably straightforward.[22]

[22](Bengtson *et al.* 1984a) defines the interpretation of (Newman 1983).

Lesson 15 MOW *is based on the Weltanschauung that organisation and work consist of a mixture of procedures and problem solving. It is useful as a general framework for understanding office work.*

MOW is, as already mentioned, based on the theory that office work consists of office procedures and problem solving. It is therefore not surprising that MOW can be said to be based on a Weltanschauung where organisation and work is seen as a mixture of procedures and problem solving.

The Weltanschauung of MOW is useful for understanding offices in several ways. It is: easy to use, sufficient, and it provides more insight than actually modelled. First, it was easy to identify office processes, tasks and procedures in the bank. The following extract from the diary illustrates this.

> Private Loan is divided into:
>
> - Customer conversation
> - Information retrieval
> - Evaluation
>
> This gives problems as conversation with the customer also takes place in the two latter processes; i.e. our understanding is not reflected in this. We therefore divide into:
>
> - Loan application
> - Evaluation
>
> which we think is a better choice of name and a better division.

This does not illustrate that the modelling was without problems but rather that it is not at all difficult to fit the real world into the models.

It was the project group's experience that the understanding expressed in the models was sufficient. By 'sufficient' they mean, on the one hand, that they had an opportunity to express a common ground of understanding with the bank clerks and, on the other hand, that they had themselves gained enough insight in order to provide constructive ideas in the subsequent design.

The Weltanschauung views an office as consisting of procedures and problem solving. The office models express only procedures, and that fact is probably the most important feature of MOW. It made the project group aware that, all the time, they were making incomplete models because they could not and should not model the problem solving aspects.

Furthermore, it made them understand the problem solving implicitly in the sense that many discussions about the bank clerks' work were in terms of different ways of combining procedures and tasks and thereby solving office problems.

MOW is not really a methodology and Newman has not proclaimed it to be a methodology. But it does have a modelling language (though it is somewhat loosely defined by Newman).

Lesson 16 MOW *supports the modelling of offices from three different but related and useful standpoints. The users understand all the models and they are able to be constructive with models taking the standpoint of the individual. However, it requires analytical skills to build all the models.*

MOW provides a modelling language for building three different models: process, task, and procedure.

The process model proved to be a model made from the standpoint of an outsider relative to the office work or looked upon from 'above' disregarding: 'who is doing what', resources, etc. The process model was particularly relevant for the systems developers because it showed what is done in the office without showing how it is done. The model provided an overview of the office work which the systems developers as outsiders did not initially have.

In building the task model, the standpoint of the insider, of one who is actually working in the office, was taken. By taking this standpoint the office work was modelled by means of units that appealed to the bank clerks and fitted well with their pre-conception of their work. The task model was, not surprisingly, useful for the bank clerks. The task model provided much more insight into *how* the work is done than the process model.

The procedure model proved to be more detailed than the two others. By modelling in terms of procedures a third standpoint was taken.

In was the project group's experience that MOW supports modelling from three distinctly different and useful standpoints. The standpoints were found to be related or connected in the sense that it is possible to move back and forth between the models without loosing track of which processes, tasks, and procedures describe the same part of the office.

The models are understood by users and can be used in a dialogue with the users about their work. The following extract describes a meeting between the project group and two of the bank clerks where a process model was discussed:

6. We had question marks in the places where we were uncertain about something. These questions were answered.
7. The modifications were added directly to the model.
8. The process name NOTICE OF WITHDRAWAL confused them because of the descriptive text. They thought CHANGE OF CONDITIONS was better and they also added two [new] processes: NOTICE OF WITHDRAWAL BY THE BANK and EXCEPTIONAL REPAYMENT OF LOAN.
   ...
9. Vincent [a bank clerk] noticed after having seen the models: "You really get a good overview of what we are doing."
10. We went through the models with Theo [a bank clerk]. It was done sentence by sentence and several in-depth explanations were given about the need for security and several types of guarantee.

The process model was mostly used by the project group to check their own overview of the loan business. It was a useful tool for this because the bank clerks actually could check the model. The task model, on the other hand, seemed to be easier for the bank clerks to understand. Hence, they could not only check the task model, they were also able to construct parts of it themselves after having seen a few examples. It is not surprising that it was easier for bank clerks to comprehend what was expressed in a task model than in a process model since the task model takes the standpoint of the individual.

The project in the South Branch was special in the sense that two approaches were used simultaneously (or at least in the same project).

Lesson 17 JSD *and* MOW *support each other.* MOW *supports analysis based on an organisational Weltanschauung and* JSD *supports design based on a technical Weltanschauung. The insight gained by applying* MOW *is useful when applying* JSD *and vice versa.*

Previous lessons have already pointed out that MOW is based on an organisational Weltanschauung and that JSD is based on a technical Weltanschauung.

MOW was used for analysis of the existing loan business. It was the project group's experience that MOW was useful in this analysis, cf. Lesson 16, and that it provided substantial insight into loan business. The insight gained was useful when designing in two ways. Firstly, it was fairly easy for the systems developers and the bank clerks to point out

the areas of loan business that could potentially be improved by the use of a new computer-based information system. The analysis by means of MOW was useful from a design point of view because it was possible to set the agenda for design through the analysis. Furthermore, the office models provided an easy and operational way of finding the boundary of the computer system because many of the entities inside the computer system had already been described in the office models as tools in the task model or as actors, i.e. the names of entities in the JSD model came directly from the office model.

Secondly, the actions and attributes of which an entity consists have already been described in the office models. One of the ideas for improvement was to computerise a report sheet used for describing a loan and the background on which it was either granted or rejected. The report sheet was specified as the entity REPORT and many of its actions and attributes stem from the process and procedure model. For example, depending on the size and complexity of a loan application it is granted or rejected by: a bank clerk, the branch manager, or the Credit Head Office. The (real world) report sheet is sent back and forth between these until a decision is taken; this is described in the procedure model and partly in the process model. The corresponding actions in the REPORT entity are: MOVE TO BM, GRANTING BY BM, MOVE TO BM, etc. The attributes of REPORT is basically the same as the contents of a report sheet. The contents of a report sheet are described both in the process model and in the procedure model. In this way the analysis by means of MOW provided a useful background for designing by means of JSD.

While designing by means of JSD much effort was still used on analysing the existing loan business. The focus and attention on few but well-defined ideas for improvement made it necessary to redirect and detail some of the office models. At the beginning, the primary aim of analysis is to find out which areas need to be improved, but later in the project, after having decided which parts to improve, an in-depth analysis needs to be done. It was the project group's experience that the simplicity of the Initial Model made it very easy to find out where to go into detail with the office models. The usefulness of the insight gained by using JSD when using MOW is also visible during analysis of consequences. After having specified the computer system in the modelling language of JSD, MOW was used to analyse and find out what the consequences for loan business would be if the computer systems were to be taken into use. The consequences were modelled by modifying the office models. The modifications were added in hand-writing to a copy of the models already made to high-light the changes. Based on the JSD

models it was fairly easy to see where the modifications should be made and what they should be.

There was thus an intimate relationship between the use of JSD and the use of MOW, an interaction where the use of one depended on the use of the other. In this sense they supported each other.

About Combining Methodologies

The above lessons apply specifically to JSD and MOW. The following lessons, however, are about combining methodologies generally.

Lesson 18 *It can be efficacious and efficient to use combinations of methodologies in the same project.*

Lesson 17 showed that JSD and MOW being based on different Weltanschauungen were useful to each other. Another way of formulating this is that the combination of JSD and MOW was useful. The combination was useful because an interaction or interplay between the use of the two could be established. It was argued in Lesson 17 that they supported each other in a direct and almost explicit way and that they did this even though they are based on two distinctly different Weltanschauungen. It may be added that the combination of JSD and MOW is useful because they are based on different Weltanschauungen. The interplay between the two Weltanschauungen is established in order to learn about the organisational aspects of the situation while designing the technical aspects and to learn about the technical aspects of the situation while analysing the existing and the future organisational aspects.

In general, the usefulness of a combination of methodologies will depend on which methodologies are combined. But if the methodologies are based on different Weltanschauungen it is likely that a project can benefit from the interplay between the involved Weltanschauungen. Such an interplay may emerge when a Weltanschauung is able to question another Weltanschauung or when the use of a Weltanschauung makes it necessary to know about the insight gained from using another Weltanschauung. For example, while using W_1 it is questioned by W_2 but is at the same time dependent on the insight gained through W_2.

The usefulness of the combination of JSD and MOW can be qualified further: the combination was both efficacious and efficient. It was efficacious in the sense that in order to analyse and design information systems it provided sufficient and proper means. It was efficient in the sense that it did not take very much time. Both these measures are relative. The point is, however, that the combination of JSD and MOW

can be said to be sufficiently efficacious and efficient if the project group by its own criteria assess it.

In general, a combination of methodologies need not be both efficacious and efficient; but it is possible to find combinations that are. As a consequence of this it is not a good idea to assume that all combinations are equally useful. It is useful to combine methodologies based on different Weltanschauungen; but with Lesson 18 it is qualified further. Thus, it is possible to find combinations of methodologies that are useful without being efficacious and efficient, and it is possible to find combinations that are both efficacious and efficient.

5.5 Methodologies at the Project Level

We can see the three projects as representatives of three modes of using methodologies at the project level:

- Following a single methodology.
- Theoretical support.
- Combination of different methodologies.

The following two lessons apply to the differences between these three modes.

Lesson 19 *It is better to use a combination of methodologies based on different Weltanschauungen than to use a single methodology.*

Lesson 7 pointed out that it is not feasible to rely on the use of a single methodology strictly. There will always be relevant aspects of the situation which cannot be captured by that methodology. In Lesson 8 it was argued that the use of a methodology will impose the inherent Weltanschauung on the systems developers' understanding of the situation and on the communication with the users. In Lesson 17 is was stated that the combination of JSD and MOW was useful. In the tradition of action research it would be legitimate to state that it is useful to use a single methodology just as it is useful to combine methodologies. However, the limitations to the usefulness will make a difference and they allow for distinction between degrees of usefulness.

There are severe limitations to the usefulness of a single methodology. For example, SA/SD cannot cope effectively with organisational aspects as it is based on a technical Weltanschauung; and TC cannot cope effectively with technical aspects as it is based on an organisational

Weltanschauung. The same objection cannot be raised to the same extent against a combination of methodologies. A technical methodology can be supported in organisational aspects by an organisational methodology and vice versa.

A combination of methodologies is not only useful to overcome some of the limitations of the Weltanschauung, especially the product-oriented aspects, of a methodology; it is also a useful construct for overcoming some of the process-oriented aspects of a methodology.

Lesson 20 *A fully operational methodology is not sufficiently flexible and a theory is not sufficiently precise. A useful balance can be achieved by combination.*

SA/SD and JSD are methodologies that can be called 'fully operational' meaning that they provide an extensive set of techniques. In the balance between 'whats' and 'hows' of a methodology, see Definition 2 on page 36, SA/SD and JSD are much more strongly built in the 'hows' dimension. Such methodologies are not sufficiently flexible. This was illustrated by the use of SA/SD in the North Project where it proved to be almost impossible (or at least beyond the capabilities of the project group) to adapt SA/SD to a situation where issue-based work is prime. On the other hand, Lessons 11 and 12 showed that a theory is so flexible that it lacks preciseness. The same can be concluded about MOW though it is a theory not in the same sense as TC; but it is certainly not as operational as methodologies like SA/SD and JSD.

The use of the combination of JSD and MOW is then an example of a working practice supported by the non-flexible preciseness of JSD and the imprecise flexibility of MOW. This is a useful balance because the overall framework is flexible but it does at the same time provide techniques that can give the sometimes-needed preciseness in modelling.

Combination of methodologies is an outcome in this thesis. The following two lessons give a more detailed answer to this.

Lesson 21 *Combinations of methodologies should support and balance: (i) technical and organisational Weltanschauungen, (ii) problem definition and problem solving, and (iii) individual and general standpoints.*

Methodologies are combined in order to achieve something which cannot be achieved by a single methodology alone. It may very well be that a single methodology does not cover essential aspects of information

systems analysis and design. Case 2 provides three pairs of aspects which should be covered by a combination of methodologies. There may be other essential aspects of information systems analysis and design which should ideally be covered, but these three pairs of aspects are those that can be learned from Case 2.

The technical Weltanschauung of JSD and the organisational Weltanschauung of MOW did, as already mentioned several times, create a useful interplay between view-points. SA/SD is based on a technical Weltanschauung. The technical focus made it difficult for the North Project to get substantial insight into the organisational aspects of loan business, cf. Lesson 8. TC is based on an organisational Weltanschauung. The East Project never got very far in finding out how the loan business of the bank could be supported by a computer-based information system. They used a lot of effort to understand the existing organisation. All in all, a combination of methodologies should include both a technical Weltanschauung and an organisational Weltanschauung and not only one of them.

Both SA/SD and JSD take a formulation of the problem as given and they give advice on how to solve the problem. This is useful if the problem is known and the solution to the problem is a specification of a computer system. Both methodologies provide substantial techniques for going from a formulation of the problem to such a specification. On the other hand, TC and MOW take no problem as given (as far as it has to do with information systems). On the contrary they can be used to find out what could potentially be taken to be the problem to be solved by a computer-based information system. Both the problem definition aspects and the problem solving aspects are necessary in a combination of methodologies.

MOW supports modelling from a individual standpoint and from an general standpoint by the task model and the process model respectively. This was useful (and necessary) because the individual standpoint provided the users with models from the viewpoint of work (and not management) as a background for considering information systems and eventually enabled a discussion of consequences of a new information system for the work. It was also useful because the general standpoint provided the systems developers with an overview that is necessary in intervening into an organisation and necessary if a new information system is to be more than a personal tool. A combination of methodologies should in general support analysis and design from an individual and from a general standpoint because of the nature of information systems and their analysis and design.

A project should seek to cover these three pairs of aspects when combining methodologies. It would be ideal if they could always be covered, but that is probably too much to hope for. The pairs of aspects can then be seen as a statement about an ideal combination of methodologies.

As pointed out in the previous lesson, there are different aspects which should (ideally) be covered by a combination of methodologies. Not all Weltanschauungen are, however, equally easy to combine.

Lesson 22 *Weltanschauungen of methodologies in combination may interact by co-existence or competition.*

JSD and MOW are based on different Weltanschauungen, but they support each other and co-exist, see Lesson 17. They co-exist in the sense that the different Weltanschauungen imply that there is disagreement between the Weltanschauungen which may or may not be peaceful. JSD and MOW co-exist as it is possible to utilise the disagreement the interplay is showing. In the case of JSD and MOW the co-existence was quite peaceful.

If, on the other hand, we were to combine JSD and SA/SD, their Weltanschauungen would be in competition. They would be in competition because the Weltanschauungen are very close to each other, they are both technical and they both take their starting point in a formulated problem. They compete in the sense that they are seeking to provide insight into exactly the same aspects of the situation, i.e. what should be the design of the computer system. The methodologies are different and their Weltanschauungen are slightly different; but to use them both is almost the same as trying to perform an activity twice.

Throughout this chapter there has been a focus on the differences in working practice and how these can be explained by the use of different methodologies and theories. The following lesson is important because it draws attention to the existence of other possible explanations.

Lesson 23 *Differences in outcome of a project are not only due to methodological differences, they are also due to other aspects like differences in the systems developers' proficiency in adapting methodologies.*

The systems developers in the North Project tried to do exactly as advocated in SA/SD, almost at all costs. They did not explicitly try to adapt SA/SD to their situation. As a consequence they struggled for a long time with the peculiarities of SA/SD, trying to fit the real world

into the models. The systems developers in the East Project had severe difficulties in adapting TC to the situation. Even though they used considerable effort in adapting the theory they never fully succeeded in establishing a useful working practice. The systems developers in the South Project adapted JSD by supplementing it with an informal description of functions and screen lay-outs. They also found a suitable working practice based on MOW and were able to combine it with JSD. The conclusion that the South Project had more success than the two other projects is not difficult to reach. The relative success is not only due to differences in the methodologies. The methodological differences do make a difference, of course; but it is relevant to look for other explanations. There are many possible explanations, e.g. differences in skills, experience, values, personalities and users. The one lesson that can be learned from Case 2 is about differences in level of proficiency applied by the projects. The level of proficiency in using and adapting is different in the three projects and can account for some of the relative success.

5.6 Summary

I have taken an extreme process-view throughout this chapter. I have not mentioned the products of the three projects, I have not compared the descriptions made; but I have compared the analysis and design efforts in The Royal Bank.

Many of the lessons learned are about SA/SD, TC, JSD and MOW showing their usefulness in different areas of analysis and design. The consequence of these lessons are taken in Chapter 7 where the details of information systems methodologies are compared based on the lessons. Other lessons are of a more general type. They apply to the use of methodologies in projects in general. These lessons are about the usefulness of using a single methodology, a theory, or a combination of a methodology and a modelling language.

Based on practical experience it has been argued that the differences in working practice are due to the differences between the specific methodologies and the differences between the general ways in which methodologies are useful. But the following contradiction summarises some of the limitations of the lessons learned.

Contradiction 2 *The main methodological contradiction at the project level is between differences in methodologies used and differences in proficiency of the systems developers in using and adapting the methodologies in a particular situation.*

This contradiction does not disqualify the lessons of this chapter; but it brings about an awareness of how far they can be generalised.

6

The Organisational Level

This chapter is about the learning and use of methodologies at the organisational level. It is based on Case 3, an inquiry into how the DP Department of the Provident Merchant Bank introduces and learns methodologies.

Section 6.1 presents the setting in which the inquiry took place, the themes investigated and the research approach applied. Section 6.2 reports on the course of action in the inquiry both as an intervention and as a piece of research. The lessons that can be learned about methodologies at the organisational level fall into two themes. The first theme is about the learning of new methodologies in a DP department and it is discussed in Section 6.3. The second theme is about methodological choice at it is discussed in Section 6.4. A summary is provided in Section 6.5.

6.1 The Inquiry

The inquiry in the Provident is an intervention into how to improve the methodological learning cycle in the Provident and as such it is at the same time research to gain insight into methodological learning at the organisational level.

Themes and Setting

The initial purpose of the intervention into the methodological processes in the Provident was to help the Methodology Group of the DP Department to introduce two new methodologies. The two methodologies had been developed by the Methodology Group over the last year. They had used considerable resources on studying the ideas of IM,[1] gaining experience, adjusting and re-developing their own version of it.

Some of the areas of interest for the intervention were expressed at the first meeting between me, as a consultant, and the Methodology Group:

- The methodology is to be learned by the systems developers. How can this be done?
- The success of the introduction of a new methodology to the systems developers is dependent on their motivation to learn a new methodology. How can the systems developers be motivated?
- The actual use of methodologies is influenced by the persons using it. How can uniform use of the methodology be ensured?
- In a broad sense, how can the use of the methodology be monitored?
- Learning will often take place when a methodology is used. How can learning be explicitly incorporated in the methodology and in the introduction of it?[2]

This interpretation of the situation with respect to methodologies did not last very long. It became apparent that the new methodologies could and should not be separated from other methodologies, or for that matter from other activities that the Methodology Group was undertaking.

A much more general theme came up as we learned more and more about the situation. Figure 6.1 expresses the focus we took after some time: Methodologies are being used and learned in an endless cyclic process.

At the organisational level the research is addressing the following questions:

Organisations and methodologies: How are methodologies used by organisations? What is the relationship between organisation and methodologies?

[1] (Flavin 1981) in particular. See also the short description of IM in Section 2.1.
[2] Project Charter, (Nielsen 1987a).

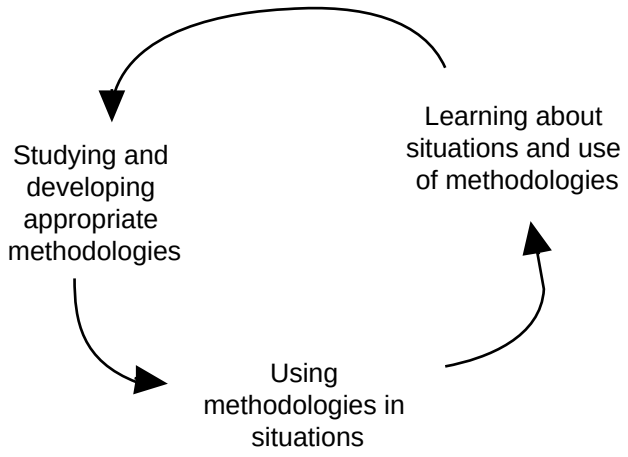


Figure 6.1: *The methodological learning cycle in an organisation concerned with methodologies. How can it be activated and operated? Key: →: leads to.*

Methodological reasoning: What methodological reasoning applies at the organisational level?

SSM as a Research Approach

The inquiry at the organisational level was conducted using Checkland's SSM as a research approach. SSM— being a methodology for intervention and learning in problematic situations in organisations with the purpose of alleviating some of the problems perceived — is an adequate methodology for dealing with the problematic situation in the Provident.

SSM is itself developed through action research where lessons have been learned about it in hundreds of projects. By using SSM in the Provident it would be along the same line of thought if we learned something about SSM from this. I have used it to learn about methodological reasoning at the organisational level (as well as about SSM as a methodology for methodological reasoning).

Checkland makes the point that the methodology can be generalised while the content of the models developed while applying the methodology only can be generalised under certain circumstances for specialised purposes.

> ... it is the methodology which is generalized rather than the content of models. There is nothing in principle wrong with general models, at least in certain situations of a well-defined kind,

> but part of the strength of systems thinking lies in the analyst's power to select a viewpoint he considers relevant and to denote systems whose boundaries do not coincide with organizational boundaries, the latter being in the final analysis, arbitrary. This is a strength not to be given up lightly, this ability to make models specific to an individual situation not only in terms of the specific values of the variables but also in terms of the classes of variables included.[3]

For the reason pointed out by Checkland, I have no intention of simply generalising the content of models. Furthermore, in terms of Schön's reflection-in-action, where situations are seen as unique, relevant models will have to be unique to the same extent. The distinction between generalising towards 'methodology' or generalising 'content of models' is, however, subtle. The content of the models that have been used in this inquiry all have to do with the learning and use of methodologies. Thus, a generalisation of the content of such models will also be a generalisation of methodology. Rather, the distinction should be between epistemological and ontological generalisations. The generalisations, which are based on the research in the Provident, are epistemological.

Having taken SSM as a research approach, in what way can we benefit from this? Even if I did research into the methodological reasoning at the organisational level without using SSM it is probably agreeable that something useful can be said about the subject. What is it then I gain from using SSM as a research approach in stead of an ad hoc approach? SSM is useful for research in the same way as it is useful for intervention. The subject (or situation) is seen as problematical where rich pictures are used to express the situation in as open-minded a way as is possible, different relevant views of the situation are formulated formally in root definitions and the consequences of these viewpoints in terms of activities are taken in conceptual models. The conceptual models are then compared with the situation to identify (more or less systematically) the differences and actions to improve the situation. In this way the benefits are the same as when it is used for intervention: a state of open-mindedness, systemically considering different views, and systematically investigating their consequences.

There are, on the other hand, differences between intervention and research. While in a pure intervention the root definitions will be relevant to the situation alone; when using SSM as a research approach there is the possibility that some root definitions are relevant to the general

[3](Checkland 1981, p. 257–258).

subject as well. Another difference is that the purpose of an intervention is improvement of the situation while in research it is learning (which may later be applied in another situation).

The carefulness of generalisation becomes important when the situation is no longer seen exclusively as a unique situation that needs to be improved but also as a useful case or example in a research subject from which we can learn. Careful generalisation can be achieved by first of all documenting the grounds from which generalisation is made. Here the rich pictures, the root definitions, the conceptual models and the forms from systematic comparison constitute the more formal documentation together with my own field notes, research diary and a few reports.[4] Again, careful generalisation is dependent on the soundness of the argumentation that leads from the documentation to the lessons learned.

6.2 The Course of Action

The Provident is among the five largest Danish banks. It has a number of large computer-based information systems for on-line handling of accounts and other products. The DP department develops and maintains the computer support for these information systems. About one hundred systems developers work in the DP department. They are organised in small groups each responsible for their area of the information systems. The systems developers receive advice and support from the Methodology Group regarding methodological aspects of information systems analysis and design, see Figure 6.2. The work of the Methodology Group includes: giving courses in data-flow modelling and project management, attending workshops and seminars to learn about new or improved methodologies, supporting and advising projects on how to develop computer-based information systems, and finally they define and maintain a set of standards and guidelines.

The Methodology Group spent fourteen months learning and adapting a general methodology for data modelling (DMM) to the specific needs and traditions of the bank.[5] The adapted methodology had recently been tried out in a few small projects and the group now felt that they were as ready as they could be to make all projects in the

[4](Nielsen 1988a; Nielsen 1988b; Nielsen 1988c; Nielsen 1988d).

[5]Basically it is IM. The theoretical foundation for the methodology is (Flavin 1981). Most of the ideas came, however, from an American consultant with years of experience in data modelling.

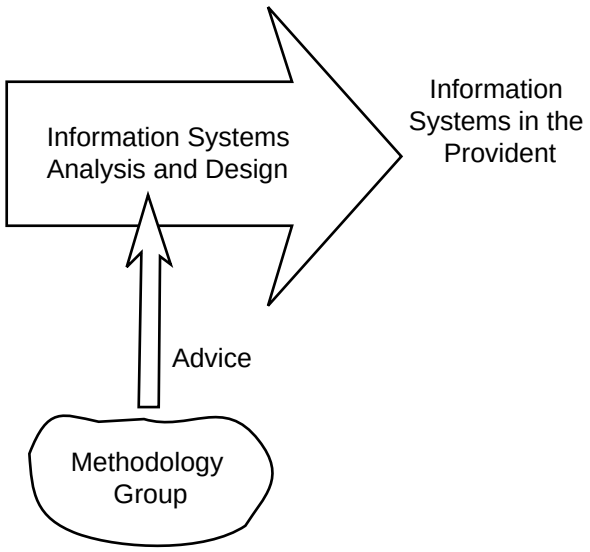


Figure 6.2: *Methodological support and advice for information systems analysis and design. The model is an application of EROS model, cf. (Checkland 1985a).*

department use the methodology. The Methodology Group perceived a variety of problems related to the methodology:

- How can the methodology best be taught to systems developers?
- Will the systems developers use it afterwards? Can it be ensured that they do use it? Should it?
- What is the role of the Methodology Group in all this?

I worked as a part-time consultant for the Methodology Group over a period of six months. The purpose of this was to help the group sort out the unstructured situation.

The rich picture in Figure 6.3 shows some of the main characteristics of the situation. The picture describes the situation as it was perceived at the beginning of the inquiry. The qualitative interviews presented in Chapter 4 provided sufficient insight to draw the picture. More profound insight was gained later, of course; but as an introduction Figure 6.3 is sufficient.

The Methodology Group has derived two methodologies: Data Modelling Methodology (DMM) and Information Architecture Methodology

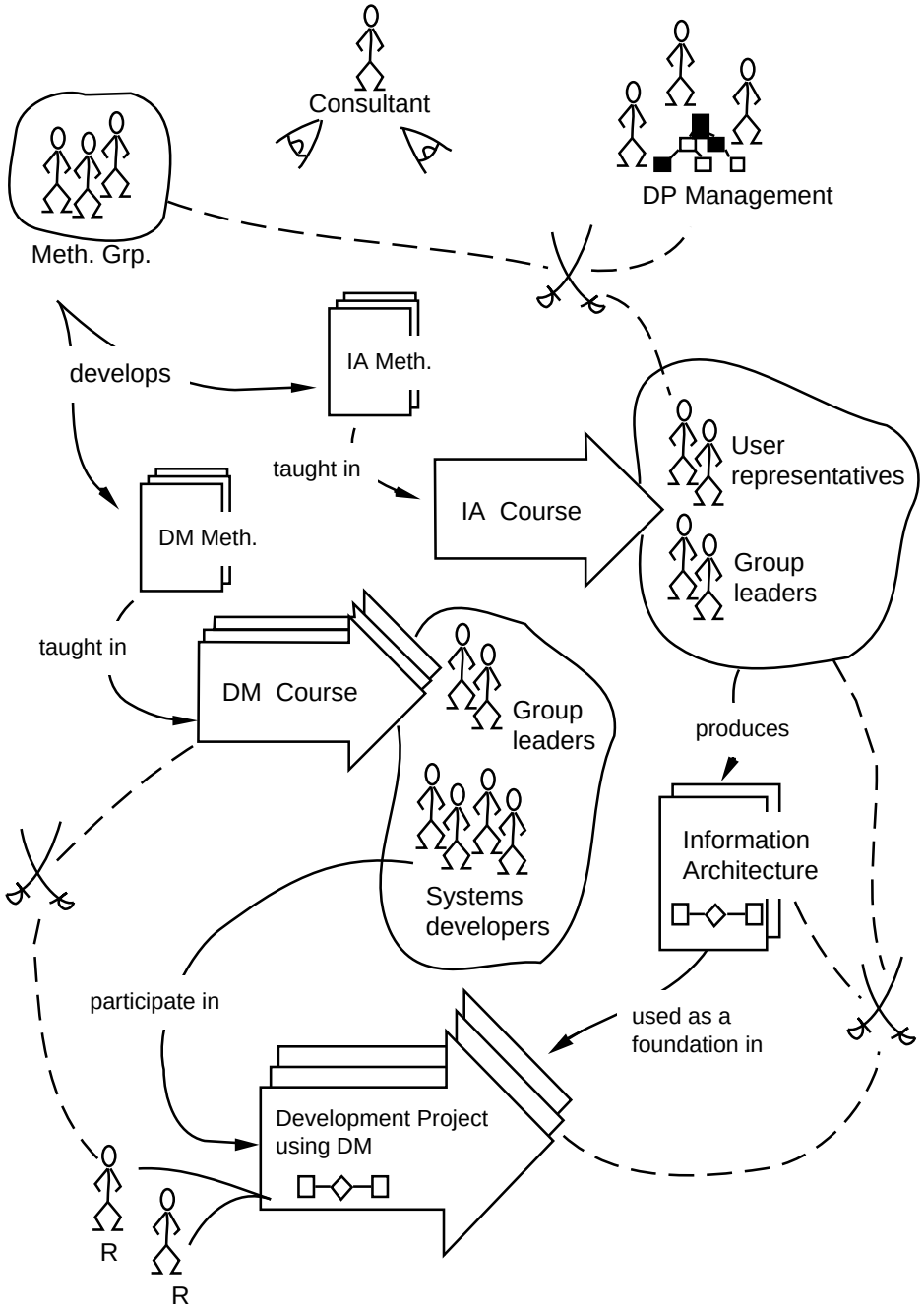


Figure 6.3: *Rich picture of the methodological situation in the Provident. The picture is explained in the text.*

(IAM). Both methodologies are based on modelling and understanding information systems as entities and relationships between entities (as in IM). The two methodologies are taught in two courses given by the Methodology Group. Systems developers and middle management (group leaders) attend the DMM course to learn the details of the modelling techniques. Users and group leaders attend the IAM course to be able to build the more general Information Architecture. The two methodologies are intrinsically related. The purpose of IAM is to let user representatives and group leaders build a single model, an architecture, at a general level for all the information in the bank. The Information Architecture is then to be used as the basis for all development projects as each project develops information systems for part of the architecture. DMM is then used in each project as a way of analysing thoroughly and in detail the models of the information already described at a general level in the Architecture.

For simplicity, only three climates are shown in the picture. First, there is a climate in the upper right corner that has to do with participation. The Methodology Group has the idea that the Architecture can only be built by letting the user representatives play an active role when using IAM. They believe that only the users have the necessary knowledge about what information the bank processes. In contrast, DP management believes that it will take too many resources to teach IAM to the users, that the users cannot be convinced to use these resources, and that the department will lose credibility because "they cannot do the job themselves." Second, the climate on the left is between some of the systems developers and DMM. During the interviews with systems developers it became apparent that they subscribe to at least two roles in relation to DMM.[6] The one role will typically refer to their work in terms of 'database design' and 'coding' while the other role typically will refer to it in terms of 'analysis', 'user involvement' and 'reflection'. The 'coding'-role has severe difficulty as DMM is a methodology for analysis in close co-operation with the users. They see no need for such a methodology mainly because DMM supports the values and norms of the other role. Third, in the middle on the right there is a climate that has to do with the confusion about the prospects of IAM.

A number of root definitions were made. About half of these were eventually used in the inquiry:

1. Information Architecture production.

[6]Chapter 4 argued that basically there are three roles: the experienced, the confident, and the professional.

2. DMM support.
4. Evaluation of introduction of methodologies.
6. Use of methodologies.
8. Adaptation of methodology repertoire.
10. Efficient analysis and design of information systems.
11. Development of quality information systems.
12. A contradiction between efficiency and quality.
13. Supervision of systems developers.
14. Limiting systems developers.
15. Communication about methodologies and experience.
16. A contradiction between supervising and limiting systems developers.
17. Standardising analysis and design.
18. Provision of methodological advice.

A few of these are relevant to this thesis and are discussed further.

As the subsequent sections will show, conceptual models were built of the root definitions and the models were used to structure several debates. Basically, there were three modes of debating: direct comparison, group activity planning, and departmental forum. During direct comparison the three members of the Methodology Group who developed DMM and IAM and the consultant systematically went through all the details of the models.[7] Group activity planning were meetings for the whole Methodology Group where the outcome of the direct comparison was summarised, discussed, and implemented. The purpose was to convey the ideas of the methodology development and to bring about the necessary change within the Methodology Group. The departmental forum was held once (at the end of the inquiry). The purpose of this was to bring together systems developers, management, users and the Methodology Group for an exchange of ideas. The starting point of the discussions was the outcome of the inquiry so far.

With hindsight, it is apparent that there were three recurring themes in the inquiry:

[7] Very much as advocated in (Wilson 1984, p. 75–82).

- *Introduction of new methodologies.* That is, how can the methodologies DMM and IAM be introduced, taught, used and maintained?
- *Methodological advice.* That is, how can the Methodology Group provide the systems developers with necessary and sufficient advice on methodological nature?
- *Methodological choice.* That is, how can a choice of methodology take place?

In the following two sections the first two themes are treated as one under the headline 'learning' and the third is treated under the headline 'choice'.

The inquiry did bring about changes: the Methodology Group changed much of their own performance, the systems developers initiated a much more careful use of their methodologies, and DP management began to support the work of the Methodology Group much more explicitly. It was mainly an inquiry that took place with the Methodology Group as prime actors. Therefore the most significant changes were of their work. However, the inquiry also provided the Methodology Group with much insight into how they could manoeuvre even in areas where the inquiry did not bring about explicit changes.

6.3 Learning of New Methodologies

The initial problematic situation in the Provident had to do with the introduction of the new methodologies. The Methodology Group had the idea that the two methodologies were appropriate to the organisation as a whole, but at the same time they expected the systems developers to try to resist a change of their methodological practice. The perceived difficulty of the introduction was the main reason for the Methodology Group to call in a consultant.

The following lesson stems from the use of the system in Figure 6.4. The root definition and the conceptual model were to some extent useful in understanding what the introduction of IAM would mean in terms of purposes and activities.

Lesson 24 *Successful introduction of new methodologies requires relevant reasoning. The arguments need to be relevant to systems developers, management, and users. The arguments need to be in terms of the organisation's methodological strategies and purposes and in accordance with experiences in using other methodologies.*

Root Definition No. 1:
A system owned by the DP Department for the Methodology Group, group leaders and representatives selected amongst bankers knowledgeable about their own part of the bank to produce an Information Architecture within one and a half years. The Information Architecture must reflect the information needs as seen by the bankers. Furthermore, the Information Architecture must as a whole be a stable but flexible framework for future projects.

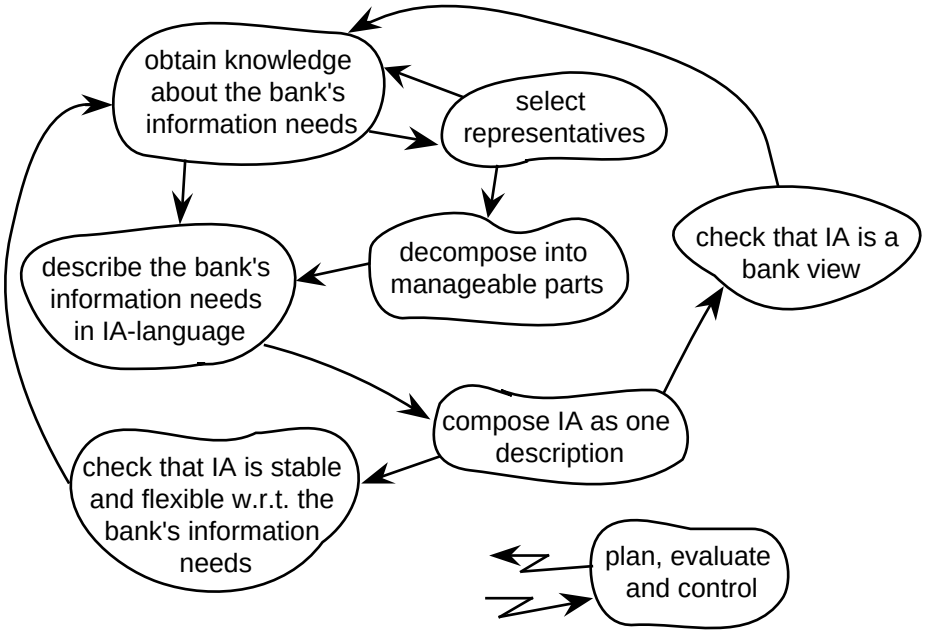


Figure 6.4: *A system to produce an Information Architecture. Key:* ◯*: activity;* →*: dependency between two activities;* ⇝*: dependency between an activity and all other activities.*

The system in Figure 6.4 explicitly deals with two reasons for introducing IAM as a new methodology in the department. First, by using the IAM properly the Information Architecture will become a bank view. Second, the Information Architecture will be both stable and flexible. They were included in the system because the Methodology Group believed that without these two criteria for evaluating the Information Architecture the system would not be worth doing.

It soon became clear that the two reasons given by the Methodology Group did not matter much to the systems developers and DP management and only partly to some of the users. None of them seemed to find the reasons sufficient for going into the huge effort of learning and using IAM. The systems developers would have to be properly introduced to the methodology since they were the ones actually going to change their methodological practice because of and according to it. The DP management would have to be properly introduced since they should authorise and allocate resources to the training of systems developers and they had the task of influencing the management of the bank to have the users to allocate bank staff to participate in the development of the Information Architecture. Finally, the users would have to be properly introduced and trained since they were going to participate in the actual modelling by means of the Information Architecture-language.

Instead, the following reasons were used in arguing the case for IAM:

1. Avoid data redundancy and inconsistency in databases.
2. Prepare data for fourth generation languages.
3. Create a basis for setting priorities and coordinating projects.
4. Put the reality of the bank rather than the computer systems in focus.
5. Create understanding and conceptual definitions shared by the DP department and the people in the bank.
6. Reduce the time used on analysis in the projects.
7. Enable computer-based information systems to be developed and changed at the same rate as the bank's information needs.

The systems developers found that (1), (2), (5) and (6) were good reasons for trying to learn the methodology. (1), (2) and (6) appealed to systems developers immediately while (5) needed some argumentation in order to become relevant.

The DP management was especially pleased by reasons (3), (6) and (7). For example, (3) means that an Information Architecture provides a view on the relationship between projects and their importance in terms of the information area of the bank.

Finally, the users found that (4), (5) and (7) were sufficient reasons for them to enter the task of producing an Information Architecture. Until that time the users had never participated in the use of a methodology. To them it was a completely new working practice and they

believed that ideally the Information Architecture should be built by the systems developers. However, they realised that if the bank rather than the computer systems are coming into the focus they, as bankers, will have to be active partakers.

It is worth noticing that different reasons are relevant to different interest groups but few reasons are relevant to more than one interest group. Furthermore, the above reasons refer to desired outcomes of using IAM. This means that the reasoning will primarily be in terms of strategies and purposes. Methodologies are thus seen as a means of achieving certain ends. Another way to put this is: It was found to be useful during the inquiry to keep the reasoning about methodologies at the level of 'why' and 'for what'.

Some of the arguments were more solid than others, and some of the arguments were more carefully examined than others. In the final consultancy report it is concluded that: "There is no warranty that IAM will actually produce the desired outcome; it is an experiment and it is thus worthwhile monitoring the use of IAM to evaluate whether the outcome is as expected."[8] Seen from a consultant's view-point this is a fair statement because all the arguments are undocumented. Seen from the view-point of the introduction of a new methodology this means that arguments do not necessarily have to be documentable in order to provide relevant reasoning for the involved partakers. The criteria of relevance can only be determined by looking at the organisation where new methodologies are to be introduced. For example, it is the users in the Provident that decided 'create understanding and conceptual definitions ...' was a relevant reason and it is the users that had to be convinced by the arguments put forward that IAM is a methodology for achieving this. Finding out what is relevant, and what is not relevant in a process of introduction is a learning process.

Lesson 24 refers to the process of introducing new methodologies to a whole organisation of systems developers based on 'relevant reasoning'. It can just as well be referred to as a political process and game where reasoning plays a significant role, but where power commodities and skills as well as possibilities for influencing the organisation are just as important.

Having introduced a new methodology in an organisation still leaves the issue of actually learning the practical use of the methodology.

Lesson 25 *Learning new methodologies is an interaction between theoretical and practical activities. The theoretical activities involve attend-*

[8] (Nielsen 1988c, p. 3).

ing courses and discussing and reflecting upon the usefulness of the methodologies. The practical activities involve experiments and use of the methodologies in order to learn from mistakes and get experience. The interaction takes considerable time and other kinds of resources.

Initially, the Methodology Group had the idea that the systems developers should be taught the DMM in a three-day course given by the Methodology Group. The course was given to a group of twenty systems developers a few months before they were expected to use DMM in the projects. The course was well-planned and gave the participants both an overview and insight into the details of DMM. The form was a mixture of lectures and exercises providing the systems developers with a variety of aspects of the methodology. During lectures and exercises much time was spent in discussing the usefulness of the methodology in terms of when and where to use it. At the end of the course it was evaluated by the participating systems developers and the Methodology Group. The course was successful in the sense that much had been learned during the three days it lasted.[9]

Soon after, it became clear to the Methodology Group that the course and the discussions were insufficient. The systems developers had not learned DMM to a desirable degree. The missing part was, not surprisingly, the practical aspects of actually being able to use it. The following root definition describes one of the systems used in the inquiry to find out what to do about the practical aspects.

Root Definition No. 3:
A system owned by the DP Dept. for the Methodology Group to assist and support the projects in adapting DMM to the needs in the projects and in using DMM under the constraints that it is a short-term support and that the projects themselves are responsible for the use of DMM.

By taking this system to be a relevant view-point and taking it through a comparison with the situation in the Provident many of the practical aspects of learning a methodology surfaced and were acknowledged. This system particularly helped the Methodology Group understand what their role should and could be with respect to the practical matters of DMM. By having the theoretically minded members of the Methodology Group engaged in practice as supporters for the system developers and vice versa, an interactive process was established where theory and practice are equally important.

[9](Nielsen 1988c, p. 8).

This interaction takes time because first the systems developers attend the three-day course and then they try to use the methodology by themselves. They will in many cases be confused or in doubt about their use of the methodology and call in a member of the Methodology Group to support them in using it. It takes time to learn a new methodology, but it also takes many other resources, e.g. a project team's attention away from problems with the information system to problems with the methodology.

The above systems view has already indicated the content of the next lesson.

Lesson 26 *In the introduction and learning of methodologies it is useful to have methodological advisors supporting and supervising systems developers. For each piece of methodological advice it is useful for the methodological advisors to understand the organisational support associated and how it contributes to quality and efficiency.*

While considering the system:

Root Definition No. 18 A system owned by the DP Department for the Methodology Group to convey advice and knowledge to systems developers about information systems development in order to improve their skills in handling efficiently the development process and ensure high quality of the product. It is assumed that methodologies can support the handling of that.

it became clear to the Methodology Group that the notion of having someone conveying relevant advice and knowledge about methodologies is useful. It may be implemented in various ways. In the Provident it was done by having the Methodology Group consisting of knowledgable people undertaking the activities of the system while the new methodologies were introduced and learned.

The Methodology Group experienced three degrees of organisational support that relate directly to methodological advice.

1. A piece of advice is an organisational standard. It must be followed by the systems developers.

2. A piece of advice has proved useful before. The systems developers will typically have to argue its irrelevance not to follow it.

3. A piece of advice is a possibility. The advisors will typically have to argue its relevance for the systems developers to follow it.

The knowledge about the associated organisational support was relevant for the Methodology Group to possess because they then knew much more about how to try to influence the systems developers by providing methodological advice. Such knowledge allowed them, for example, to be proper advisors instead of simply acting on the management's behalf.

It was found during the inquiry that it is useful to think of this relationship between individual advice in terms of the contradiction between efficiency and quality. A contradiction with respect to methodologies is seen as a totality of two *opposite* aspects. One of the systems we looked at was 'a system to develop information systems efficiently' and based on that it is possible to debate which aspects of methodological advice contributes. Another system was 'a system to develop information systems of high quality' and based on that it is possible to debate which aspects of methodological advice are contributing to the quality. Both these systems are relevant in methodological inquiries — the efficiency-system mainly from a management view-point and the quality-system mainly from a user view-point. But the systems developers (and their advisors) who live under the requirements of both management and users will have to consider both these aspects despite the fact that they are in contradiction. Efficiency and quality are contradictory measures which means that they can seldom be fully fulfilled at the same time by the same methodological advice. Choices will have to be made in order to achieve a reasonable trade-off and balance between the two.[10] Giving methodological advice is not just conveying a piece of knowledge about methodologies. If the advisors seriously want to help the systems developers they must confront themselves with the situation of the systems developers and hence be able to relate different advice to the specifics of the situation and to each other. From time to time there is a need to relate advice to each other.

6.4 Methodological Choice

The concept of *methodological repertoire* proved during the inquiry to be useful in the understanding of the process of choice of methodologies in a particular situation. The issue of methodological choice grew into one of the most significant issues in the work of the Methodology Group. It

[10] The use of explicit thinking in contradictions combined with the use of SSM is discussed in (Mathiassen and Nielsen 1989; Nielsen 1989a).

turned out that we did not address the issue directly by formulating a system for choosing methodology, but rather it was addressed by looking upon it from various angles.

Lesson 27 *The methodological repertoire of systems developers is limited. A repertoire can be developed, but it is necessary to build up substantial experience to make it useful. A repertoire needs to be continuously modified and adapted to the systems developers' and the organisations' needs.*

The three arche-typical systems developers referred to in Chapter 4 are, as already mentioned, systems developers in the Provident. The interviews with these three and some of their colleagues leaves the impression that the repertoire of methodologies of a systems developer is very limited. Each systems developer had at best heard of ten methodologies and is at best able to use five of them.

This observation is supported by the fact that it is a difficult and slow process for systems developers to learn methodologies like DMM and IAM (which are not very sophisticated), cf. Lesson 25. To internalise methodologies like DMM and IAM, i.e. to make them a part of a systems developer's repertoire, takes time and other resources. It takes a considerable number of years to obtain a voluminous repertoire. Furthermore, it is unlikely that a lot of resources will be put into the extension of a repertoire with new methodologies that are close to those already contained in it. Additionally, the inclusion of new methodologies in a repertoire is likely to push some old ones out. All in all, the methodological repertoire of systems developers is very limited.

The system outlined in Figure 6.5 was used extensively in learning about the repertoire. The system is far from being in operation in the Provident; but the notion of this system did provide a most useful debate about its prospects. It is based on a fairly rationalistic Weltanschauung where it is assumed that the nature of methodologies and repertoires is fully tangible, and that bringing all information together will inevitably lead to a decision on the adaptation of methodologies and repertoires. This was noted already in the beginning of the discussions about this system:

> A lot of resources have been used on the development of DMM and IAM, but other methodologies are used as well in the department. There are, in fact, many methodologies available. It is therefore interesting to see what is needed to create a repertoire of proper and cohesive methodologies.[11]

Root Definition No. 8:
A system owned by the DP Department for the Methodology Group to adapt the repertoire of methodologies for systems developers to ensure a repertoire of proper and cohesive methodologies as a basis for an awareness on which methodologies should/could/may be used in a project.

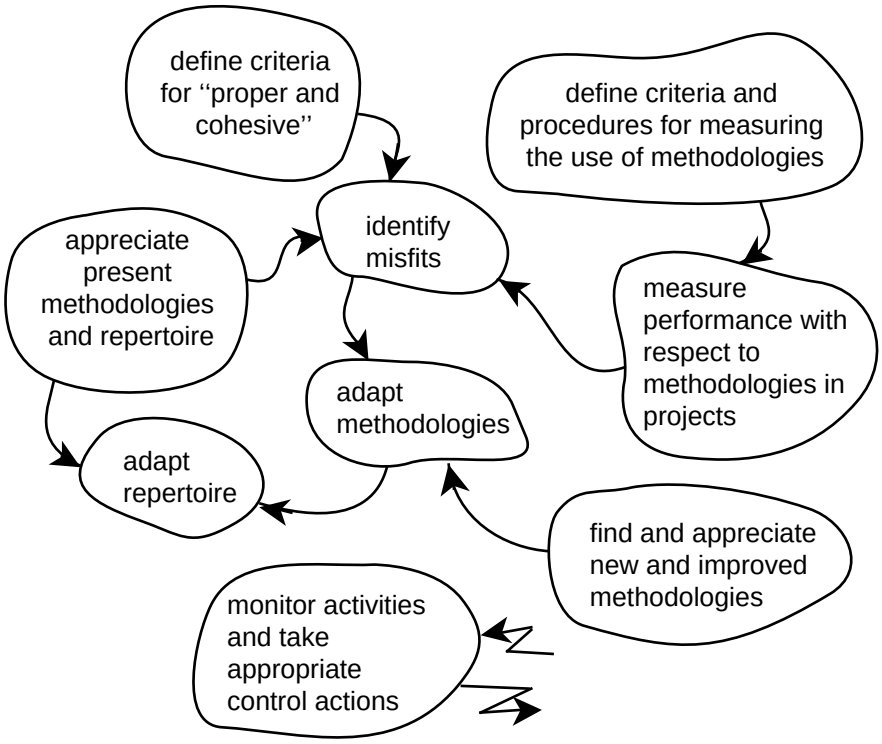


Figure 6.5: *A system to adapt methodological repertoire. Key: ◯: activity; →: dependency between two activities; ⇝: dependency between an activity and all other activity.*

It came as a surprise to the Methodology Group when they realised the difficulty and slowness of bringing DMM and IAM into the systems developers' repertoire. The whole process of introducing and learning DMM and IAM to all the systems developers (and some of the users) of the Provident is a way of doing the activities of the above system. Looking

[11] (Nielsen 1988c, p. 17).

upon all this, it seems feasible to develop a repertoire even though it is the repertoire of a hundred systems developers. But as Lesson 25 pointed out it is necessary to do more than introducing the methodologies and carry out the theoretical parts of the learning (which of course is not itself trivial in any sense). It is necessary to gain experience to make the methodology a usable element of their repertoire.

Just as methodologies are adapted to a particular situation whenever they are used, so is a repertoire adapted and modified when it is used. Each systems developer has only one repertoire and the development, modification and adaptation of a repertoire is a longitudal process. A repertoire cannot be chosen, it can only be slowly modified. This makes it even more important to be aware of whether the repertoire is up to date and meet the needs of the systems developers and the organisation. The process of introducing and learning the two methodologies in the Provident was initiated by a desire to keep up with the changes of the bank and utilising new methodologies in doing so.

Lesson 28 *A choice of methodology in a particular situation is guided more by repertoire and personal preferences than by what, from a rational viewpoint, seems to be optimal.*

In the Provident it was at one time asked why a methodology like SSM was never used at least in the early stages of a project. The answer given by one of the old systems developers is simply: "Nobody knows what it is." In a particular situation SSM would be an ideal and optimal methodology for defining what a new information system could be about. It is probably an optimal choice, but from the viewpoint of the involved systems developers it is not at all ideal or optimal unless it is already an integral part of their repertoire. The systems developers in a project or the methodology staff in an organisation may well have different repertoires. A choice of methodology is most likely to be a choice amongst the methodologies of the repertoires present.

This chapter has in several ways already addressed the issue of 'the nature of the process of methodological choice'. The following lesson draws upon the other lessons and makes the nature of the choice explicit.

Lesson 29 *The process of choice of appropriate and adequate methodologies is a learning process related to a specific organisation involving both practical use of methodologies and reflections upon the use.*

It is not a surprising lesson after having been through all the other lessons. It had to end like this. The above lesson is not less important just because it has become evident.

The repertoire influences the process of choice significantly, cf. Lesson 28. The process of bringing new methodologies into the repertoire is itself a learning process involving the practical activities of using the methodology and the theoretical activities of discussing and reflecting upon the use, cf. Lesson 25.

It is, however, not necessary to turn to the previous lessons. The whole inquiry reported in this chapter is a itself learning process. Some of the major characteristics of this learning process are, as SSM was used, that it was closely connected to the specific situation in the Provident, and that it involved both use of methodologies (DMM and IAM) and reflections upon the use.

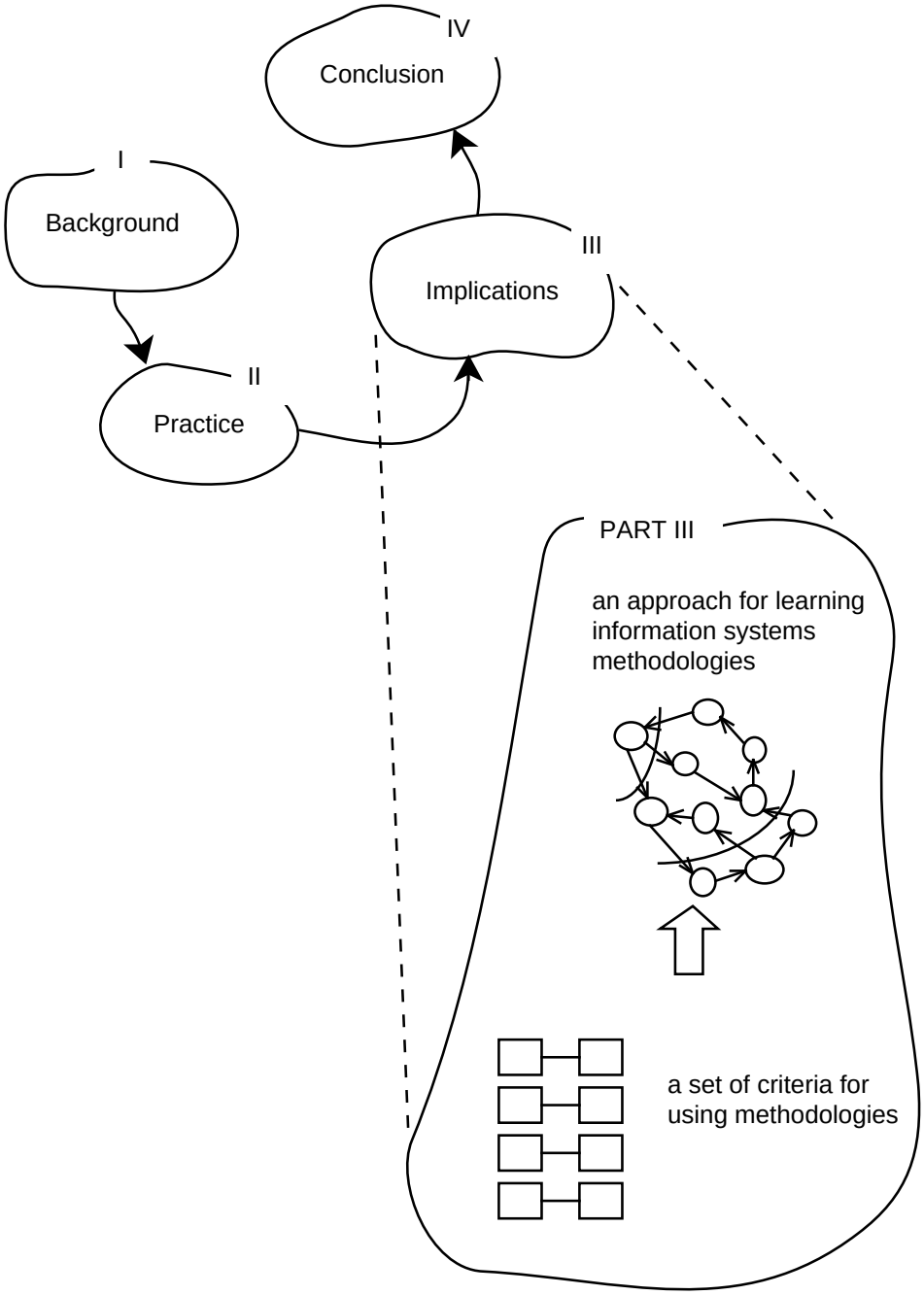
6.5 Summary

This chapter has reported on an inquiry into the learning and use of information systems methodologies at the organisational level. The lessons learned from this are not restricted to the organisational level; they just happened to surface easier here than at the project level. The first three lessons deal with the conditions and possibilities for introducing and learning new methodologies. The last three lessons deal with the nature of methodological choices and the importance of understanding the repertoire.

It underlies all the lessons in this chapter that the organisational process of changing methodological practice is the same as in any form of organisational change in terms of resources demanded and time consumed. It should—in principle—only be done when there are good reasons. This is summarised in the following contradiction.

Contradiction 3 *The main contradiction at the organisational level is between, on the one hand, the established tradition in the development organisation with respect to use of methodologies and, on the other hand, the perceived need for change of the use of methodologies.*

Part III
Implications


IV
Conclusion
I
Background
III
Implications
II
Practice
PART III
an approach for learning information systems methodologies
a set of criteria for using methodologies

7

Criteria for Using Information Systems Methodologies

This chapter provides an overview and comparison of information systems methodologies. This is done by reformulating the lessons learned about the specific methodologies in Chapter 5 (and to some extent Chapter 6) as a set of criteria for using methodologies.

Section 7.1 is an introduction to this chapter explaining the nature of the criteria and how they reflect the lessons in Part 2. Section 7.2 deals with the criteria related to the domain of use, Section 7.3 with the criteria related to the conditions for use and, finally, Section 7.4 with the criteria related to the consequences of use. The criteria are discussed and the chapter is summarised in Section 7.5.[1]

[1]An earlier version of the contents of this chapter has also been published in (Nielsen 1989b). Some of the criteria have been reformulated to serve the purpose and terminology of this thesis; the main difference is that I do not any longer distinguish between operational and managerial users. Also, the examples used are more closely connected to the methodologies referred to in this thesis and I have eliminated references to the Tool-approach (Ehn and Kyng 1987; Bødker *et al.* 1987), Office Automation Methodology (OAM) (Sirbu and others 1984), decision support systems, e.g. (Sage *et al.* 1983), the methodology by Ehn & Sandberg (1979), and instead of referring to all prototyping approaches as one I have taken STEPS as a modern example.

7.1 Introduction

The criteria for using methodologies are divided into three groups based on three fundamental questions about information systems methodologies:

α: *When can the methodologies be used?*

β: *What is needed?*

γ: *What are the implications?*

The questions relate directly to three areas of concern.

α: The α-question is about the inescapable characteristics of the situation at hand in which the methodology can be potentially useful. The methodologies are related to different domains where they can be used or will be useful. α is called the domain of use.

β: Even though a methodology might fit the domain of use it requires a range of skills and attitudes in order to be potentially useful. The skills and the attitudes are called the conditions for use.

γ: Methodologies have different impact and lead to different working practices. The γ-question relates to what impact a methodology has, what are the implications and the consequences of a methodology. Hence γ is called the consequences of use.

Figure 7.1 shows the criteria.

The criteria stem from the lessons learned with SA/SD, TC, JSD and MOW reported in Chapter 5 (and to some extent from the experience with SSM in Chapter 6). All the criteria can be traced back to lessons learned from actual use of the methodologies.

Each criterion is intentionally thought of as a distinction that separates a whole in two contradictory aspects or as a spectrum where the end-points are contradictory. This is partly based on the old observation that it is not possible to have 'day' without 'night', or to have 'life' without 'death', etc. It is the same with methodologies; it is inconceivable to attach the assessment to SA/SD that it is useful in contexts characterised by procedural activities, as in Lesson 4, page 84, without having some conception of what are *not* procedural activities, i.e. problem-solving activities. The criteria are not all contradictory in a strong dialectical sense; but the two concepts that make up a criterion are opposites.

Each criterion is presented in terms of three aspects:

α: Context of Use

procedural task v. problem-solving task
personal task v. collective task
production task v. administration task
greenfield area v. existing information system

β: Conditions for Use

Stance towards Inquiry

harmony perspective v. conflict perspective
given problems v. problematical situations
observation v. intervention

Systems developers' Qualifications

technical skills v. organisational skills

Users' Qualifications

sources for inquiry v. partakers in inquiry
non-analytical v. analytical

γ: Characteristics of Use

Working Practice

model making v. structured debate
analytical inquiry v. experimental inquiry
analysis v. design
computer systems v. information systems
process v. structure

Models

abstract models v. concrete models
overall models v. detailed models
formal models v. informal models

Figure 7.1: *Criteria for using information systems methodologies.*

Concepts: The concepts that make up the criterion are explained briefly.

Origin: The origin of the criterion (i.e. what lessons can they be traced back to?).[2] For some of the criteria it is necessary to relate to some of the existing literature in the field and to give proper credits.

Example: Some examples of how different methodologies map onto the criterion. The experience gained with SSM during the project in the Provident has shown that it is indeed a methodology different from most other methodologies. SSM (and thus also ISM, cf. Section 2.1) will therefore serve as examples in many of the criteria.

7.2 Domain of Use (α)

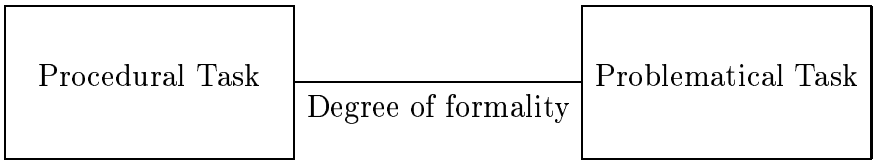
The domain of use of a methodology is the area of the real world in which it is potentially useful, cf. Definition 3 in Section 2.3. Some methodologies claim that they can handle any domain of information systems. Other methodologies are more modest and relate to a specific domain, e.g. offices. Newman's MOW is designed specifically to handle office automation, while JSD on the other hand relates to management information systems.

Concern about information systems is expressed in relation to tasks in the organisation where the inquiry is being conducted. It can be a primary task, i.e. the business of the organisation, or a minor task as part of or in relation to a primary task. A task is, for example, the production of a newspaper or the handling of loans in a bank. All tasks are basically about human activity, though at different levels of abstraction.

Procedural v. Problematical Task

A task can be either procedural or problematical. A procedural task is governed by strict rules for what should be done, but most important how it should be done, i.e. whether a procedure exists. A methodology in this category will focus on the procedures for the task, not what is actually done. Complementary to a procedural task is a problematical task, i.e. no procedure exists for the task and the persons doing the task are therefore finding out what to do and how to do it by themselves. Methodologies useful to approach problematical tasks acknowledge that human activity is more than following procedures.

[2] All the lessons of Part 2 can be found in Appendix A on page 191. As much reference is made to the lessons it will be easier to see them there rather than in different parts of the Chapters 5 and 6.



From Lesson 4 we know that SA/SD is useful in domains characterised by procedures and routine activities and that it is not suited for domains characterised by problem-solving. The criterion originates from this lesson because we have here found a methodology that can handle procedures and routines in a consistent and useful way. In terms of tasks: SA/SD is useful in domains characterised by procedural tasks and it is not useful in domains characterised by problematical tasks.

Lesson 13 reveals that JSD, on the other hand, is useful in domains characterised by procedures and problem-solving. Likewise, MOW deals explicitly with both procedures and problem-solving, according to Lesson 15. These two lessons are important for two reasons. First, there exist methodologies that can handle problematical tasks. It is therefore relevant to have a criterion that distinguishes between methodologies like SA/SD on the one hand and JSD and MOW on the other hand. Second, JSD and MOW deal with problematical tasks in two different ways. JSD does it implicitly, e.g. by modelling a loan in a bank without considering the problematical *work* necessary to establish a loan. MOW does it more explicitly by modelling the procedural aspects of the tasks in a way that makes it possible to see how the procedures may be used in more problematical tasks.

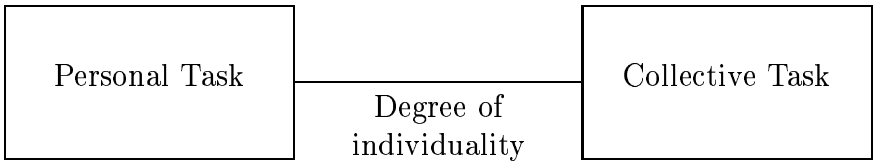
The methodologies map onto the criterion as follows:

- SA/SD: procedural tasks.
- JSD: procedural and problematical tasks, implicit.
- MOW: procedural and problematical tasks, explicit.

Personal v. Collective Task

A task can be either personal or collective. A personal task refers to human activity related to a single person or a group of persons doing exactly the same. Methodologies for inquiring into personal tasks are focusing on what individuals are doing and how it is done. Complementary to this, the collective task is referring to human activity as the combined effort of a group of people. The focus here is on what the group is doing

and how it is done. Methodologies in this category are often investigating collective tasks by focusing on how each individual contributes to the task and how links between individuals are established, maintained, and used.



The distinction between personal and collective tasks arises most directly from Lesson 9. It says that TC is based on the Weltanschauung that organisation and work can be seen as transactions governed by contracts. A transaction is an exchange of something between two parties, i.e. it is a coordinated collective action. In this way, the notion of transaction cannot be used in analysing and designing for a personal task. Methodologies like MOW, on the other hand, deal explicitly with personal tasks in one of the three models used, cf. Lesson 16. The distinction between personal and collective tasks is thus useful because it distinguishes methodologies. They do not fall into two disjoint categories rather the criterion provides two extreme and opposite aspects of tasks in the domain of use.

Methodologies map onto the criterion in the following way:

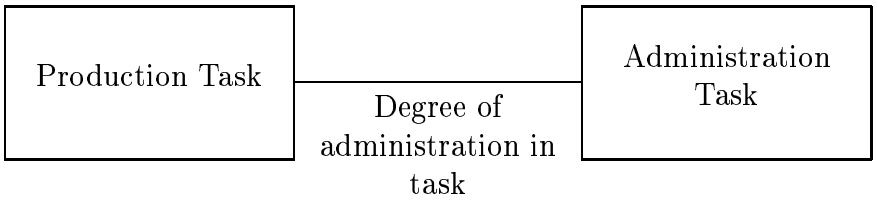
- TC: collective tasks.
- MOW: personal and collective tasks.
- ISM: collective tasks.

ISM is an example of an approach oriented towards the collective tasks, because it is mainly based on the development and use of a primary task model. The aim of the methodology is therefore to look at human activity in wholes that do the business of the organisation rather than looking at the individual being and acting in the organisation.[3]

[3](Wilson 1984, p. 245). I refer to the making of the primary task model in Stage 2.

Production v. Administration Task

Tasks can be separated into production and administration. Administration has been associated with office work and it is often a useful metaphor in this connection. Administration is here seen as the complementary to production, e.g. the production of a newspaper or parts thereof. Administration is the support of the production, e.g. accounting and handling of salaries, and the sale of the products, e.g. handling subscribers for a newspaper. It is worth noticing that many organisations are without any production.



In Lesson 9 it was stated that TC is a useful framework for understanding information needs of administrative systems. TC obtains this by focusing on contracts and not as much on the transactions they govern. In terms of tasks, this means that TC is useful in domains characterised by administrative tasks. MOW is an approach towards administrative tasks by the same line of reasoning, cf. Lesson 15. None of the methodologies used in Case 2 seem to be equally useful in a domain characterised by production tasks; but as far as evidence is concerned, a methodology like JSD could in principle be useful in a production task as well. The point is, however, that TC and MOW are *not* useful in production tasks. Therefore this criterion is relevant.

As examples:

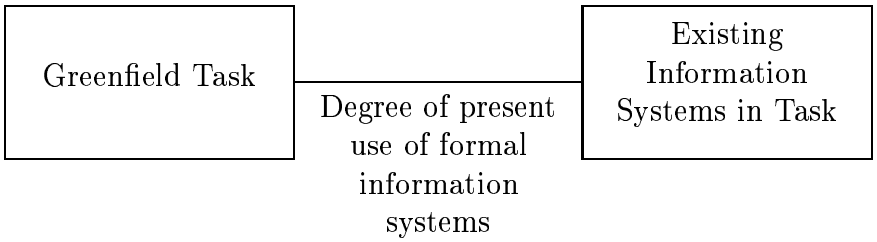
- TC: administration task.
- MOW: administration task.
- STEPS: production and administration task.

There is nothing that hinders a methodology like STEPS[4] from being useful in domains characterised by production tasks as well as administration tasks. It contains no modelling language or assumptions about the domain and there seems to be no limit to the tasks in which it can be useful.

[4](Floyd *et al.* 1989).

Greenfield v. Existing Information System

The final α-criterion is a distinction between the greenfield tasks and tasks using existing computer-based information systems. When concern about information systems is expressed in relation to tasks where computer-based or just formal information systems already are in use this introduces a whole set of problems like, for example, integration, share of data, etc. Few methodologies are able to cope explicitly with these concerns.



Lesson 13 pointed out that it is not possible by means of JSD to consider existing computer-based information systems that are not likely to be altered by the new design. In the South Project where JSD was used it was necessary to specify a computer-based information system where substantial parts of the data (attributes in entities) came directly from another information system. This was not possible by means of JSD. While using the other methodologies in the Royal, SA/SD, MOW and TC, none of these problems occurred. Thus there exist methodologies in both categories and the criterion arises because it distinguishes between them.

JSD and ISM are examples of methodologies where the extremes of this criterion are highlighted.

- JSD: greenfield task.
- ISM: existing information systems in task.

ISM, by the device 'The Maltese Cross', handles a number of existing information systems, their internal relations, overlaps and gaps in the information provision, etc.[5]

[5](Wilson 1984, p. 221–224).

7.3 Conditions for Use (β)

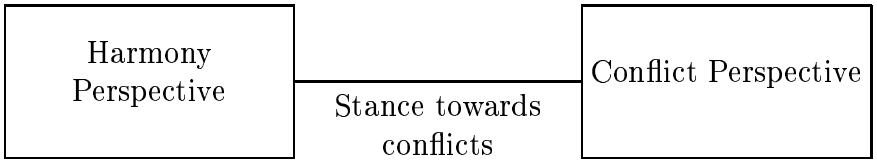
The conditions for use of a methodology are the requirements to the actors' skills and attitudes. The conditions for use can be divided into two groups: stance towards inquiry and the needed qualifications of the actors.

Stance towards Inquiry

The stance towards inquiry is the attitudes, beliefs and perceptions of the analysts and users about the situation and the inquiry.

Harmony Perspective v. Conflict Perspective

The harmony perspective expresses the view that the relation between actors in the situation is basically harmonious. In this view an apparent conflict is seen as misunderstandings and the conflict will disappear if more relevant information is provided. Furthermore, the actors are seen to share a purpose in their activities. The conflict perspective on the other hand expresses the view that the relation between actors is fundamentally conflictual and that conflicts cannot disappear, they can only be less obvious.



JSD assumes harmony because it only requires *one* model to be built, i.e. the right model, cf. Lesson 14. It does not explicitly allow for disagreement about what should be modelled and how it should be modelled. SA/SD adopts a harmony perspective as well, cf. Lesson 5. TC and MOW do not to the same extreme extent assume harmony as they both support the making of several models expressing various viewpoints. There is, however, an attempt made to make all the models built, e.g. by means of MOW, to a coherent set of models. A modern example of methodologies based on a conflict perspective is SSM where viewing a situation from different and sometimes conflicting standpoints is at the very core.

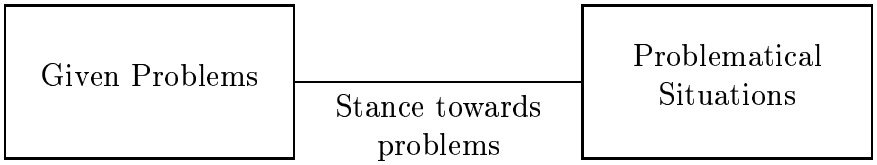
Previously, Ehn and Sandberg have suggested that the harmony and conflict perspectives are relevant in information systems development.[6]

As examples of the methodologies that assume the two opposite aspects:

- JSD: harmony perspective.
- SSM: conflict perspective.

Given Problems v. Problematical Situations

When problems are believed to be given the methodologies facilitate the specification of the problem and the finding of the solution. When problems are not believed to exist, problems are seen as perceived difficulties in the situation or it is perceived that things could be improved. Methodologies in this category emphasise the different perceptions of the situation and that problems cannot be objectively defined.



In Lesson 21 it was argued that a combination of methodologies should support and balance problem solving and problem definition methodologies. In the argumentation SA/SD and JSD were seen as methodologies that assume problems to be given. For example, JSD takes as a starting point that it is clear to the users (at least the managerial users) what the system is about, i.e. the problem is given, and JSD supports the specification of the solution.[7] TC and MOW were, on the other hand, seen as methodologies that assume that problems are not at all given, i.e. what is called 'problematical situations'.

The discussion of the difference between assuming that problems are given and that situations are problematic where problems are perceptions of the situation are numerous.[8]

[6](Ehn and Sandberg 1979). The approach by Ehn and Sandberg provides a model for information systems development based on the assumption that employers and employees have conflictual purposes, hence it adopts a conflict perspective.

[7]See also (Jackson 1983, p. x–xi).

[8]To name a few: (Checkland 1981), (Lanzara 1982), (Schön 1983), and (Wilson 1984).

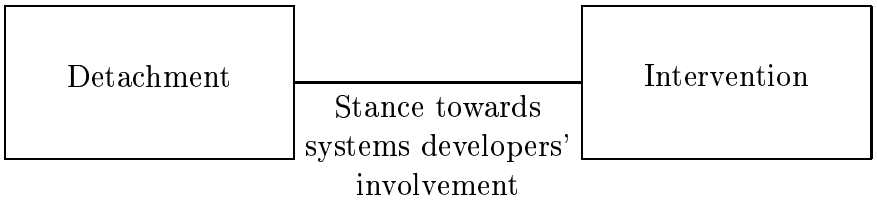
When mapping methodologies onto this criterion it turns out that all methodologies fall under only one of the aspects, never both.

- JSD: given problems.
- MOW: problematical situations.
- ISM: problematical situations.

SSM and therefore also ISM do not assume that problems are given.[9] These methodologies have their genesis in the disbelief in given problems and they mainly support problem definition.

Detachment v. Intervention

The systems developer can be thought of as an observer who sees the content of the situation from outside. The actions of the observer do not influence the situation, and based on the observations some recommendations are passed on to the users (or clients). In this way the systems developer is detached from the situation. On the other hand, the task can be thought of as intervention into the situation where the actions of the analyst cannot be separated from other actions taken in the situation with respect to the impact on the situation. In intervention the analyst is seen as an actor who learns about the situation by acting in it.



This criterion comes about because of the perceived differences between the methodologies used in Case 2 and Case 3. JSD assumes, for example, that the systems developer acts as an observer because it prescribes a minimal interaction between the systems developers and the situation—be told about the needed design and then deliver the required specification. SSM proved to be a methodology of a wholly different nature where intervention is primary. Learning is brought about in a situation because of the intervention, i.e. the desire to improve the situation. ISM, being based on the same set of ideas as SSM, provides

[9] (Checkland 1981) and (Wilson 1984).

sufficient examples for the necessity to distinguish information systems methodologies according to the criterion.

The distinction between detachment and intervention is also due to Schön. In his work on reflection-in-action, cf. Section 3.2, it is argued that intervention as opposed to observation is one of the main characteristics of reflection-in-action.[10]

- JSD: detachment.
- ISM: intervention.

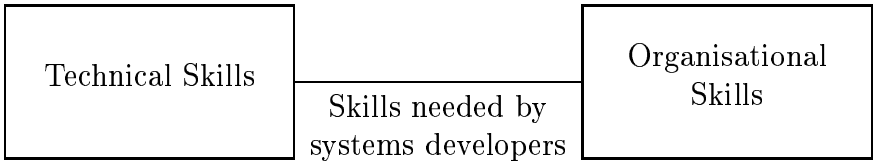
Systems Developers' Qualifications

Different methodologies demand different skills from the actors, and the skills denote, for example, the ability to undertake analytical and abstract thinking.

Information systems development involves two kinds of actors that are here seen as two roles: the systems developer and the user. This is a very simplistic model of the actors involved, but a more thorough account can be found elsewhere.[11] The systems developer is here seen as the one that brings about improvement in the situation with respect to the information systems in question, i.e. the systems developer is the main actor in the thinking about the content of the situation. The users are the actors in the content of the situation.

Technical v. Organisational Skills

It has been argued that information systems development requires both technical and organisational skills of the systems developers. The technical skills are about computers, data processing and structuring, etc. The organisational skills are about information, business procedures, human activities and interaction, etc. Methodologies are associated with these skills because the use of methodologies requires basic skills beyond the knowledge about the stages of the methodologies and how to do these.



[10] (Schön 1983, p. 163).
[11] Cf. (Land and Hirschheim 1983).

From Lesson 21 it is known that combinations of methodologies should support and balance technical and organisational Weltanschauungen. SA/SD and JSD are both based on a technical Weltanschauungen, cf. Lesson 4 and Lesson 13, while TC and MOW are based on organisational Weltanschauungen, cf. Lesson 9 and Lesson 15. In order to use SA/SD and JSD it is therefore necessary for the systems developers to have technical skills and likewise necessary to have organisational skills to use TC and MOW.

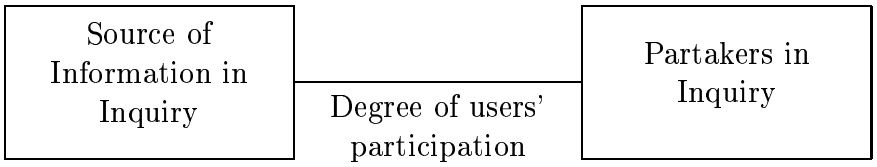
As examples:

- JSD: technical skills.
- TC: organisational skills.

Users' Qualifications

Sources for Inquiry v. Partakers in Inquiry

Users can participate in the inquiry in two ways. Either as sources of information for the inquiry in the sense that they are only to pass on the content of the situation to the systems developers. Or they can be partakers in the inquiry in the sense that they are directly involved in the inquiry. Many methodologies are based on the view that users possess knowledge about the content of the situation that must be elicited and digested by the systems developers. A few methodologies see users as the main element in the inquiry and facilitate a (more or less) full participation by the users. Methodologies that further the involvement of the users have often been called "participative" approaches.



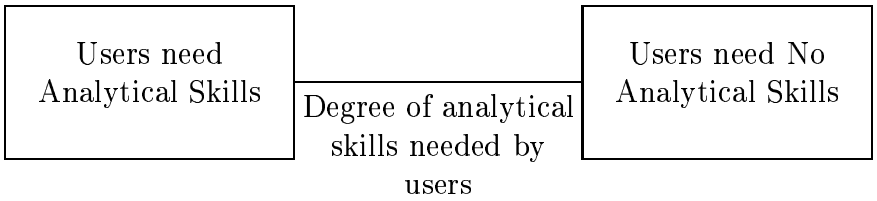
Lesson 6 stated that SA/SD sees systems developers as model builders and users as model checkers. This means that the systems developers elicit the domain knowledge from the users and describe it in a data-flow model. The users merely serve as sources of information which the systems developers can consult if in doubt about domain-specific matters. Other methodologies take the stance that the users must necessarily be partakers in the inquiry. In the inquiry in Case 3 SSM did require concerned actors to be partakers.

- SA/SD: sources of information.
- SSM: partakers.
- STEPS: partakers.

The main idea of STEPS is participation, cf. Section 2.1, page 26.

Non-Analytical v. Analytical

Methodologies are based on different kinds of modelling languages, some of which require skills to understand and take an active part in inquiry based on formal or very abstract models. Hence the distinction between whether the methodologies require analytical skills by the users or not.



In Lesson 14 JSD is found to include the building of models that are difficult for the users to understand. But JSD does not prescribe that the users should take part in the modelling. It is therefore fair to categorise JSD as a methodology that does not require analytical skills by the users. SA/SD is based totally on the data-flow modelling language and requires analytical skills of the users, cf. Lesson 6.

The examples show the variety of answers.

- JSD: non-analytical, by exclusion.
- SA/SD: analytical.
- STEPS: non-analytical, by prototypes.

STEPS is based on the use of prototypes in the parts of the inquiry where the users participate. Prototypes are simply used and require no analytical skills to be experimented with.

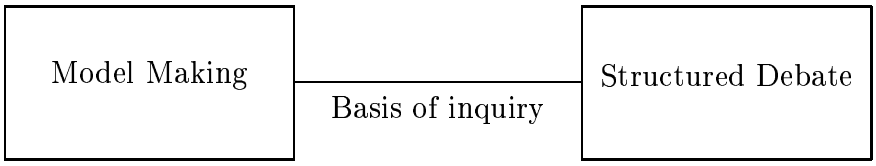
7.4 Characteristics of Use (γ)

The characteristics of use of a methodology is the thinking and acting implied by using it. These characteristics relate to the working practice the methodologies lead to and to the models that are built by using them.

Working Practice

Model Making v. Structured Debate

Two different bases of inquiry into a situation have been found: model making and structured debate. Model making is centred around the development and discussion of—in principle—one model. In model making models are often made to describe the existing organisation so the analyst can use that as a fixpoint in the development of a suitable information system. The most important characteristic of model making is, though, that one model of the proposed design is made. In the modeling process a number of models might be used, but the aim is to end up with a sufficiently useful model. Complementary to this is the structured debate where models play a minor role in relation to the aim, i.e. to debate rather than make the model. In structured debates, models are the structure or the means for debating and models often express and make explicit the different viewpoints of the actors in the situation.



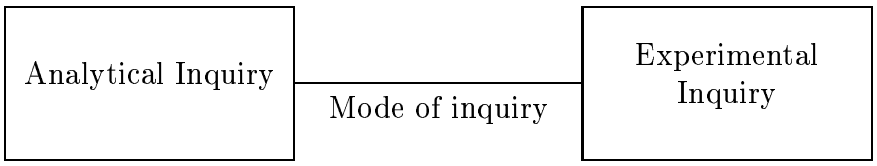
Both by using JSD and SA/SD one coherent model is made and all attention is concentrated on making this model, cf. Lesson 5 and Lesson 14. That there is another possibility than model making becomes clear when using SSM.

- SA/SD: model making.
- ISM: structured debate.
- STEPS: model making and structured debate.

SSM and ISM are classic examples of methodologies for structuring debates amongst actors in the situation. STEPS working with several prototypes for exploration purposes but ending up with basically one model (i.e. the final version of the system and its specification) is based on model making and on structured debate.

Analytical Inquiry v. Experimental

Another distinction is between the analytical and experimental mode of inquiry. The analytical mode is often associated with what has been called 'traditional analysis', and that is in most cases a useful term. The analytical mode is an approach to inquiry where thinking is prior to trying out the concrete consequences of an idea in the real world. An inquiry in an analytical mode applies a lot of thinking and discussion about abstract (paper-based) models before going into a phase where concrete models (running systems) are developed. The opposite mode of inquiry is the experimental one where relatively little effort is used on thinking before acting since the aim is to experiment with a variety of concrete models (not necessarily but typically running systems). The driving force in this mode are the experiments and their outcome.



The criterion arises because methodologies like JSD and SA/SD conduct inquiries in the analytical mode. For example, Lesson 14 pointed out that the models built by means of JSD are the basis for implementation. In other words, a paper-based design is produced and JSD does not provide guidelines as to how experimentation with this design or its implementation could take place. It is thus assumed that the thinking behind the design is sufficient for arriving at a useful information system.

SSM as used in the Provident is both analytical and experimental. It is analytical in the same sense as SA/SD and JSD, i.e. thinking before acting, but at the same time it acknowledges and utilises the insight that may be gained from confronting the models with the situation and from attempts to change the situation.

As examples:

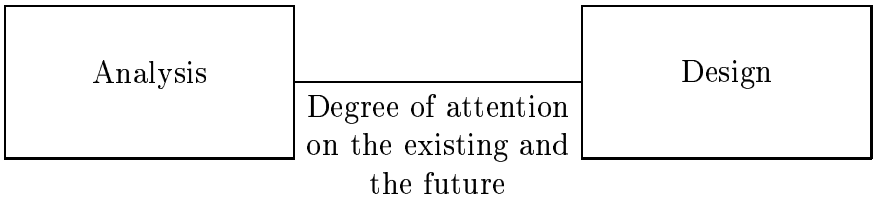
- SA/SD: analytical.

- SSM: analytical and experimental.
- STEPS: experimental.

STEPS and other prototyping approaches are very much based on the experimental mode of inquiry. The prototype is modified according to the outcome of the previous experiments.

Analysis v. Design

An important criterion is based on the distinction between analysis and design. Analysis is about understanding and interpreting the content of the situation and proposals for design. Analysis has traditionally been concerned with describing the existing organisation from a computer Weltanschauung. Design is about technical and organisational possibilities, i.e. feasible and desirable changes in the aspects focused on. Design generates visions about technical and organisational changes including descriptions of programs and computer systems. A common phrasing of this is that in analysis attention is on the existing work situation and in design on the future work situations.



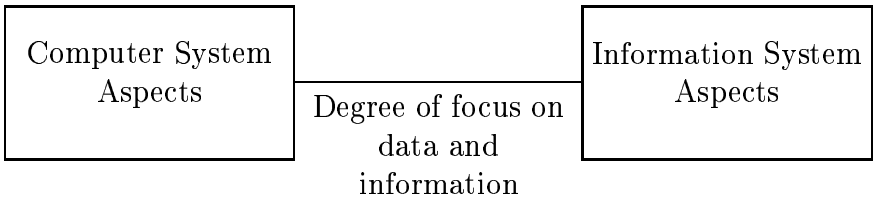
Lesson 17 has already provided an argument for this criterion: MOW is primarily providing a modelling language for analysis of the existing office work and JSD is primarily providing a modelling language and guidelines for designing a computer system. The criterion is used to distinguish these two aspects of methodologies.

- MOW: analysis.
- JSD: design.

Computer Systems v. Information systems

One of the main criterion distinguishes between data and information. Data is here taken to be whatever can be stored in and processed by a

computer. Information is then data and the associated meaning. The consequence of this is that to focus on information means to focus on how and why specific elements of data are used. 'Computer systems' means then systems concerned with data and 'information systems' means systems concerned with the meaning and use of data.

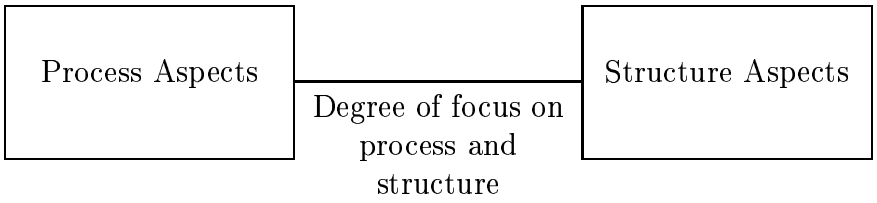


SA/SD and JSD are based on computer Weltanschauungen and thus focus on data while TC is based on an organisational Weltanschauung which is useful in understanding information, cf. Lessons 4, 13 and 9.[12]

- SA/SD: computer system.
- TC: information system.

Process v. Structure

To focus on process aspects of a situation means to perceive things to be moving, developing and changing; to focus on structure aspects of a situation means to perceive properties of the situation to be steady or stable. Aspects of both process and structure will be present in most methodologies. The criterion is therefore applied to distinguish between the primary aspects.



[12]This distinction has also been pointed out in (Wilson 1984, p. 247–255) and in (Checkland 1981, p. 168).

JSD focuses primarily on structures. This is due to the basic elements in the initial model being entities and relations between entities, cf. Lesson 14. The procedural behaviour of an entity is also modelled, but the prime concern is the structural aspects. SA/SD focuses primarily on data processes. The data-flow models express, by means of the bubbles, data processing and the relation between them, by means of the arrows, cf. Lesson 5. The structures that the data-processes work on, i.e. data-dictionaries, are also modelled, but the processes are primary.

Wilson refers to the same distinction as whether a methodology's modelling language is in terms of verbs or nouns.[13]

- JSD: structure.
- SA/SD: process.

Models

According to Mathiassen, distinctions between the abstract and the concrete and between the overall and the detailed are necessary.[14] It is suggested that *moving* from the abstract to the concrete and back (similarly with overall and detailed) is the important aspect of the distinction. As criteria it is more interesting to assess the primary aspects used in or expressed by the modeling language in the methodologies and by that provide a basis for thinking about the qualities of the methods.

Models are abstract if they express concepts and relations between concepts. Concrete models on the other hand have an appearance that do not reveal the underlying conceptual thinking.

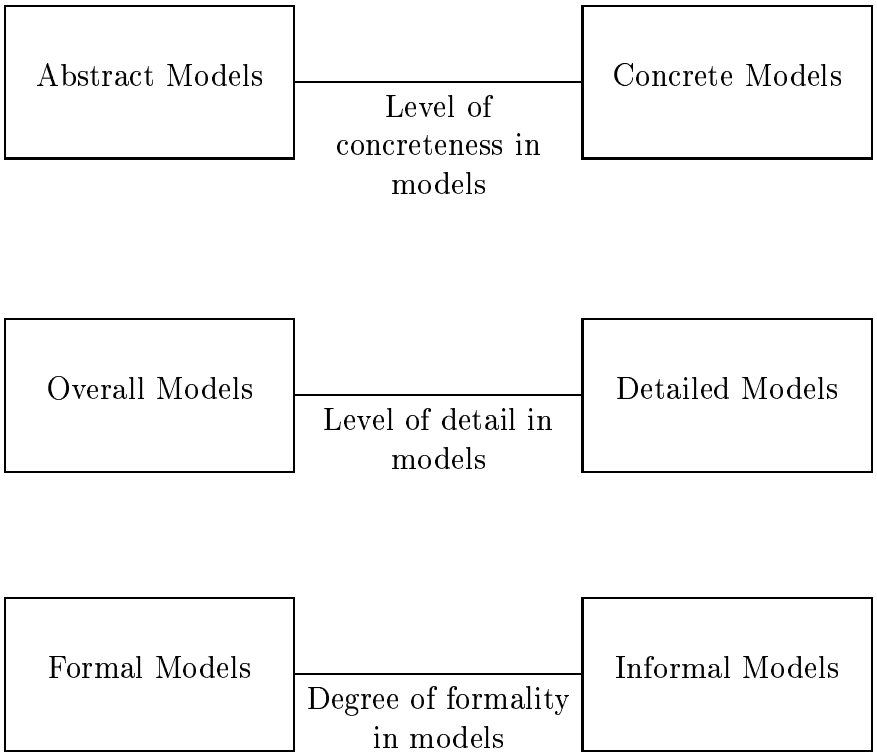
The overall model expresses insight relevant to the whole of the task or domain around which the inquiry is centred. A detailed model expresses only parts of the task or domain in question.

The distinction between formal and informal models is based on the observation that models of computer systems eventually still have to be formalised to fit them into a computer. A formal model is a model that potentially can be run on a computer. An informal model is then the opposite which means it cannot in the same sense be run on a computer or it does not make sense to try to.[15]

[13] (Wilson 1984, p. 247).

[14] (Mathiassen 1981, p. 111). Mathiassen also distinguishes between moving from language to phenomena and from phenomena to language, but that assumes model making as a basis of inquiry and is therefore not used here.

[15] (Mathiassen and Munk-Madsen 1985).



JSD starts out with models of details in terms of entities; from these detailed models an overall model is made, cf. Lesson 14. JSD uses mainly formal models since already from Stage 2 the modelling language is basically a programming language (the model describes a computer system, cf. Lesson 13).

The models of MOW can be categorised as both overall and detailed models. There are two kinds of office work models: those based on a general standpoint and those based on an individual standpoint, cf. Lesson 16. In the models based on the general standpoint, the office is seen from the above without paying any attention to who is doing the activities but only to the logic in what is being done and why. These models are at the overall level. In the models based on the individual standpoint, the office is seen from the point of view of an office worker; this is a detailed model because each model only expresses insight into parts of the office work.

- JSD: abstract, formal models at overall and detailed levels.
- MOW: abstract, informal models at overall and detailed levels.
- STEPS: concrete, formal models.

- ISM: abstract, informal models at an overall level.

STEPS is representative of the archetype of methodologies using models with concrete appearance, i.e. the prototype. A prototype is a formal model since it is running on a computer.

The primary task models used in ISM are informal by nature and it is not an aim in the methodology to automate the descriptions as they are only perceptions of the situation, hence there is no reason to formalise the modelling language. ISM uses primarily overall models. This is an intrinsic property of the systems approach in ISM, and even when models are developed to the very low level they still express insight from an overall point of view.

7.5 Summary

The above criteria are general in the sense outlined in Chapter 3. They are offered as criteria for using information systems methodologies in other situations than where they arose from. The criteria stem from practice and are thus relevant to consider in another situation.

The idea of having such criteria can be used in the following two ways. Firstly, the distinction between the three groups of criteria: α, β and γ is a list of useful overall questions to ask about a methodology: (α) When can the methodology be used? (β) What is needed? (γ) What are the implications? This means that in another situation where it is found useful to *extend* the set of criteria these questions may possibly give guidance as to where to look for new criteria.

Secondly, the criteria are formulated as two opposite aspects and not simply as features of methodologies. It is useful to have the criteria as two opposites because it implies a *choice* between the two aspects. The choice may be to use a methodology that supports one of the aspects, the other aspect, simply both aspects or both aspects in a certain mixture or balance, etc.

The criteria themselves can be used in the following ways. If a new methodology appears it can be mapped onto the set of criteria of this chapter. Some of the mapping can be done simply based on reading about the methodology. It is likely, however, that there will be criteria onto which it is not immediately possible to map the methodology. This calls for a use of the methodology. While using the methodology it is then important to know what to look for, though that should not restrain the awareness only to these criteria.

8

Towards a New Approach

In this chapter I offer my generalisation of the methodological practice presented in Part 2 in such a way that its leads towards an approach for learning and using methodologies. It is tempting to say that I offer a methodology: The Meta Methodology. This would, however, be too pretentious. I have neither experimented with the generalisation nor tried it in a real world setting. I therefore hesitate to call it a methodology. It is a generalisation that in due time may eventually lead to a meta methodology. Only time and further research can show that. This does not disqualify my attempt towards an approach; it merely points out its status.

Section 8.1 outlines the approach by its structure and intellectual foundation. Section 8.2 illustrates the activities and the techniques in the approach. Finally, Section 8.3 provides a brief summary of this chapter.

8.1 The Ideas Behind the Approach

It was argued in Section 3.2 that general approaches are useful but insufficient and that situational approaches are needed as well. Furthermore, it was argued that Soft Systems Methodology[1] (SSM) is a relevant intellectual foundation for a situational appreciation approach. Seen from

[1](Checkland 1981).

the view-point of methodological practice as presented in Part 2, SSM is a relevant foundation for three reasons.

Firstly, SSM is a methodology based on *soft systems thinking*. This means that it is based on the notion of human activity systems. A human activity system represents an account of a view-point on human activity. The methodology is in that sense subjectivistic, i.e. beliefs about reality rather than reality itself are in focus. There is a reason to believe that the new approach must indeed and more than anything else deal with the beliefs about and the conceptions of methodologies and their usefulness. Lesson 3 pointed out that systems developer's use of methodologies is based on subjective measures such as their experience and their association with professional values. Other lessons point in the same direction: the new approach could with good reason be based on soft systems ideas, cf. also Lesson 28.

Secondly, SSM is based on *learning*. The whole methodology embodies the idea of learning by eliciting view-points from a situation, treating these as wholes and then learning from the confrontation between the view-points and reality. The new approach could in this way benefit from being a learning approach, cf. e.g. Lesson 29.

Thirdly, SSM is a methodology for *improvement* of problematical situations. The starting point is a problematical situation and the intention all the way through is to improve the situation so as to alleviate some of the perceived problems. Problems do not exist and have a life by themselves. They are view-points possessed by concerned actors. Ideally, the new methodological approach is an approach for improving methodological matters (i.e. perceived methodological problems) by structured methodological reasoning. Hopefully, the use of the new approach will lead to improved methodological situations.

The significant consequence of taking SSM as an intellectual foundation is that the new approach *inherits* decades of research into soft systems thinking and practice. Many important aspects of general problem solving will not have to be re-learned and re-formulated. The new approach can be seen as an application of SSM to a more restricted set of phenomena than general problem solving. More precisely, it can be seen as a projection of SSM to the world of information systems methodologies. Doing this projection we need to be aware of and consider carefully the consequences. The consequences of such a projection is that in formulating and in using the approach, the subject under study, information systems methodologies, is known. This means that it is possible and necessary to assume much more about reality than is possible through the general use of SSM.

Figure 8.1 outlines the very idea of the approach. A situation in the

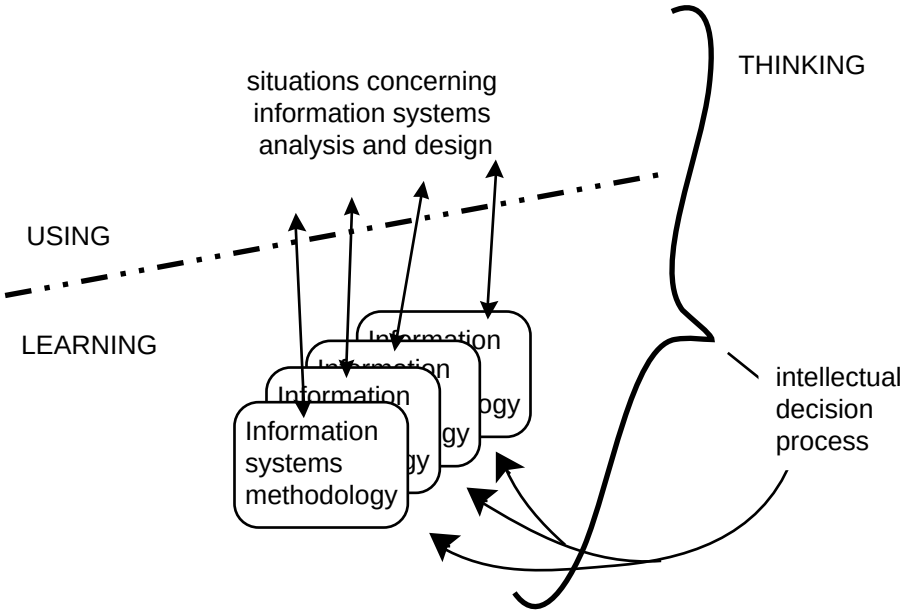


Figure 8.1: *An outline of a new approach for learning and using information systems methodologies.*

real world concerned with information systems is the fix-point of the approach. In this situation different information systems methodologies may be used. Relative to this situation the information systems methodologies belong to a meta world in the sense that they offer different ways of *thinking* about the situation. The distinction made here between the real world and the meta world corresponds to the distinction made in SSM between the real world and the systems thinking often referred to as 'above and below the line'. Because this new approach is itself about methodologies it embodies another distinction, namely the distinction between on the one side the world of reality and methodologies (seen as a whole) and on the other hand the world that is meta relative to this. The most significant difference between SSM and the new approach is that SSM operates in two modes based on one distinction while the new approach operates in three modes based on two distinctions.

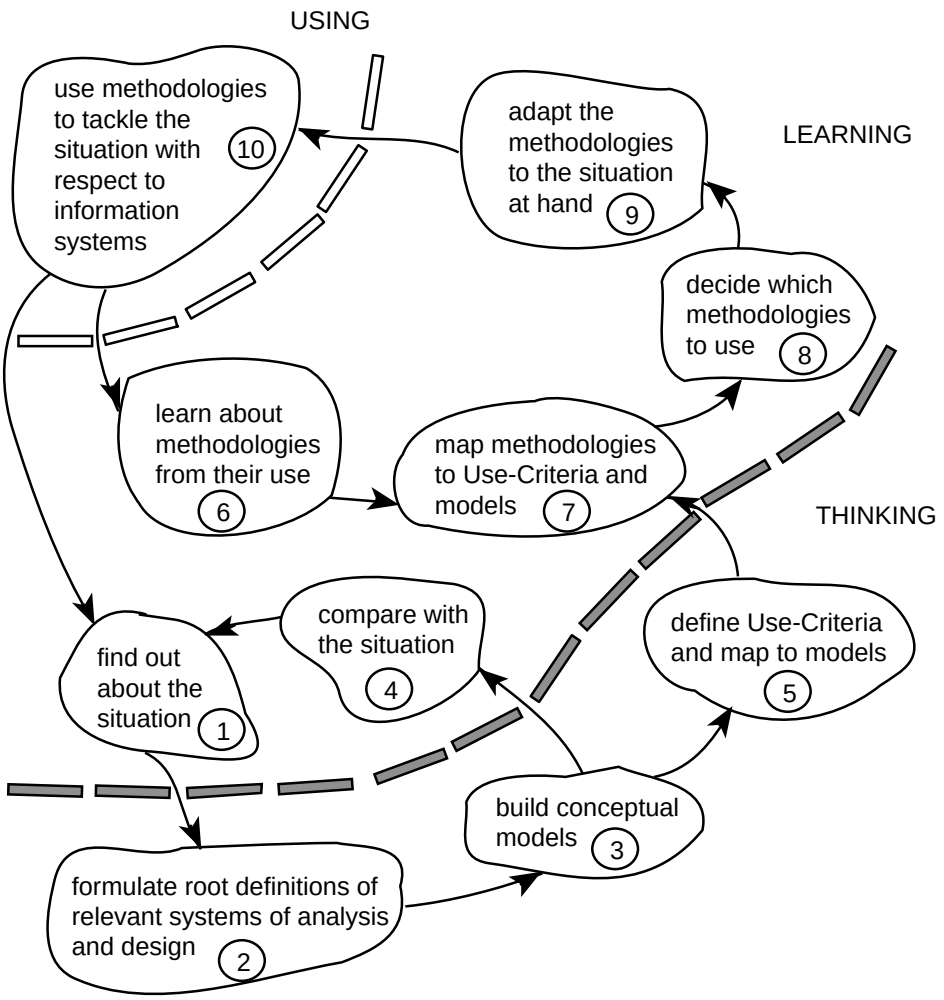


Figure 8.2: *The activities of the new approach. Key: →: logical dependency; ◯: activity.*

8.2 The Approach in Detail

Figure 8.2 shows the main activities of the approach. The intellectual foundation of the approach is, as already mentioned, the ideas underlying SSM, and some of the activities will be close in content to some of the activities of SSM. Additionally, the distinctions between different worlds already shown in Figure 8.1 are found again here making the distinctions as in SSM between 'above and below the line'.

Activity 1: Find Out About the Situation

This activity is similar in nature to the stages of SSM for finding out about the problematic situation. The established techniques of SSM may be applied when performing the activity. This involves: drawing rich pictures with elements of structure, process and climate, and Analysis I, II and III, i.e. analysis of problem owners, roles, and power commodities.[2]

If there are consequences for this activity of the lessons learned it is that Analysis II seems to be of particular relevance. In Analysis II, roles are analysed with respect to which roles can be identified in the situation and their associated values, norms and expected behaviour. Lessons 2 and 3 refer to the methodological reasoning at the individual level. The point is that there is a diversity of use of methodologies that relates to systems developers' experience and values. By analysing roles some useful insights into the easiness of changing methodological practice may potentially be obtained.

This approach is used when concern about methodologies in information systems analysis and design has been expressed or when the users of the approach have encountered problematical areas that are intrinsically related to methodological problems. That is a difference between SSM and this specialised approach. It is a more or less known area of concern otherwise this approach would not be in use in that situation. This does not at all mean that problems are taken as given; the purpose of this activity is, as in SSM, to get as rich an expression of the situation as possible. It does mean, however, that a more precise title of Activity 1 would be : 'find out about the situation with respect to analysis and design of information systems and the use and learning of methodologies'.

Activity 2: Formulate Root Definitions

Root definitions of (hopefully) relevant systems are formulated and tested exactly as in Stage 3 of SSM.[3] A root definition, being a precise description of a concrete human activity system, can be tested for its well-formedness by the CATWOE test.

A distinction between 'primary task' and 'issue-based' root definitions has been made.[4] A primary task root definition describes a system relevant for learning about the purposes and most significant activities of the organisation in question. An issue-based root definition, on the

[2] (Checkland 1981, p. 165–166) and (Checkland 1985b).
[3] (Checkland 1981, p. 166–169, 215—227).
[4] (Checkland and Wilson 1980).

other hand, describes a system relevant to learn about issues and minor activities in the organisation that are interesting only for a limited time. Checkland & Wilson provide the lesson that an inquiry should be based on both primary task and issue-based root definitions. The specialisation of this is that the primary task root definitions relate to analysis and design while issue-based root definitions relate to change of the use of methodologies, i.e. of the working practice.

Another criterion of relevance of root definitions lies in the three levels of methodological practice and reasoning. In a methodological inquiry it may be relevant to consider all three levels. Thus, root definitions that relate to the individual level, the project level, and the organisational level should be made. For example, when trying to find out which methodologies to use in a specific project it may be relevant to look at root definitions that relate to the organisational level to understand the purposes and strategies that constrain the project.

Activity 3: Build Conceptual Models

Based on each of the root definitions made during Activity 2 a conceptual model is built. A conceptual model describes the minimal set of activities and their dependencies that are necessary in order to do the transformation expressed in the corresponding root definition.[5]

Activity 4: Compare with the Situation

This corresponds to Stage 5 in SSM, i.e. 'comparison of models with the real world'. This activity is performed almost as in SSM. The focus is on arriving at a set of systems that are relevant rather than at some definition of change. If some model (and thus a system) shows during comparison that it is not as relevant a view as could be expected then this leads to some iteration where some more finding out about the situation (1) is done, root definitions are re-formulated (2), models re-built (3), and finally some more comparisons are made (4). The outcome is a satisfactory set of systems that can be utilised in the later activities of the approach.

Activity 5: Define Use-Criteria and Map to Models

Chapter 7 has shown a number of contradictory criteria for using information systems methodologies, Use-Criteria for short. These general

[5](Checkland 1981, p. 169–176, Appendix) and (Wilson 1984) for guidelines as to how these models are built.

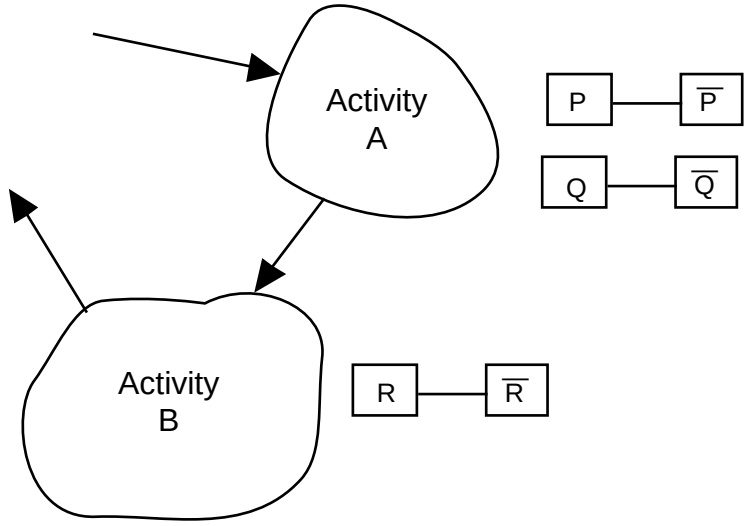


Figure 8.3: *Use-Criteria are mapped to conceptual models. Part of a conceptual model is shown. The contradictory Use-Criteria are mapped to the activities as contradictory views on potential measures of performance. Key:* $P, \overline{P}$*: two opposite aspects of a Use-Criteria.*

Use-Criteria are not necessarily relevant to all models and other more local Use-Criteria may be relevant instead. It has already been discussed in Chapter 7 how the general and the local Use-Criteria relate and supplement each other.

Figure 8.3 shows how Use-Criteria can be applied in this activity. The main idea is to think of all the available Use-Criteria, both general and local, as potential measures of performance for activities of systems models. Let us say, for example, that in one of the models used in some methodological reasoning there is an activity called: 'elicit users' conceptions of information needs'. If we now consider the Use-Criteria of Chapter 7, several of these are potentially relevant, for instance 'model making v. structured debate'. The criterion is now associated with the activity if it can be argued from the systemic outlook of the model that both aspects are proper measures of performance of the activity. In the example, this means that the criterion can be chosen to be associated with the activity if we can argue consistently that 'model making' *and* 'structured debate' can both be measures of 'elicit users' conceptions of information needs'.

This activity is a matter of:

- defining which Use-Criteria to apply, and
- mapping these Use-Criteria to the systems models.

Notice that in doing the above, the final decisions about measures of performance have not been taken. It is yet to be considered which methodologies support which measures. Such decisions will ideally have to consider measures and methodologies simultaneously.

Activity 6: Learn About Methodologies From Their Use

The outcome of this activity is, as in Chapter 5, lessons learned about methodologies in use. The mapping of methodologies to Use-Criteria, Activity 7, is obviously dependent on this. Therefore the activity must try to answer the question: 'For what is this methodology useful?' It is also from this activity that new Use-Criteria may arise, and the activity must also try to answer the more fundamental question: 'What are the concepts by which usefulness can be understood?'

Activity 7: Map Methodologies to Use-Criteria and Models

Figure 8.4 illustrates the main idea of this activity. To each of the aspects in each of the mapped-on Use-Criteria the methodologies that support are mapped. This corresponds to the examples given in Chapter 7.

Activity 8: Decide Which Methodologies to Use

The result of this activity is a decision about which methodologies should be used in this specific situation. A number of methodologies or parts thereof have been found to be potentially useful seen from various viewpoints expressed in the systems models and Use-Criteria. Decisions have to be taken as to what aspect of each of the Use-Criteria should be applied. That is, several measures of performance must be chosen. By doing this a *liaison* between the methodologies and the specific situation is established, see Figure 8.5. During Activity 7 each of the potential models has been mapped onto the Use-Criteria. Thereby, they have been associated with what could or should be done in the situation as expressed in the systems models. The technique for establishing the liaison is simple:

- For each activity it is decided which aspects of the Use-Criteria should be taken as a measures of performance for this activity.

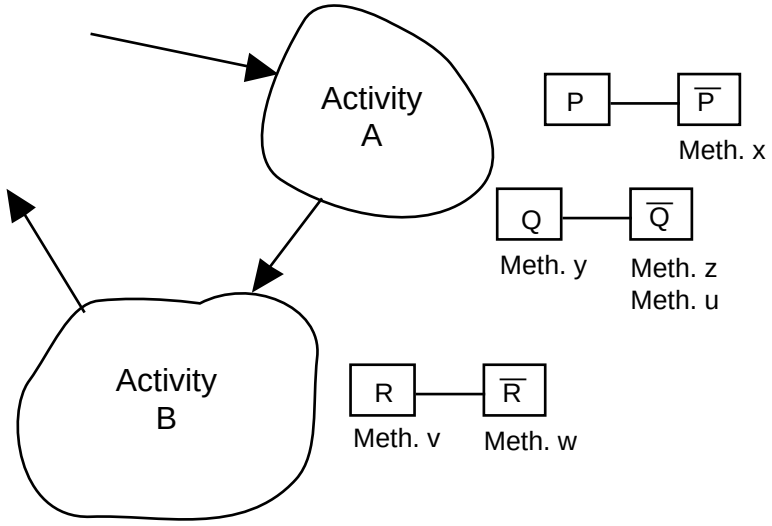


Figure 8.4: *Methodologies are mapped to the Use-Criteria.*

- For each of the chosen measures of performance it is decided which methodologies or parts thereof that should be used to achieve proper performance of the activity.

The two decisions can only be logically separated. In practice they will be hard to distinguish, e.g. 'Do I want to do this because I like the measure or because I like the methodology that can achieve it?'

This seems to be fairly trivial, but it is less obvious when the full consequence of SSM and the lessons of Part 2 are taken. First, Activity 8 corresponds to Stage 6 of SSM where change is defined according to the two criteria: systemically desirable and culturally feasible. The liaison and all parts of it will in the same way have to obey the same two criteria simultaneously. For a liaison to be systemically desirable it must be possible to argue its case from the systems models used.[6] For a liaison to be culturally feasible it must be possible to argue that it can be implemented (with some possible obstacles, but implemented in the end). Cultural feasibility can in the context of this approach be interpreted as culturally feasible with respect to the use and learning of methodologies.

The notion of *repertoire* is important in understanding what is culturally feasible. According to Lessons 27 and 28, the methodological

[6]Remember that both aspects of an applied Use-Criterion will support the overall measure of performance of the system in which it is applied.

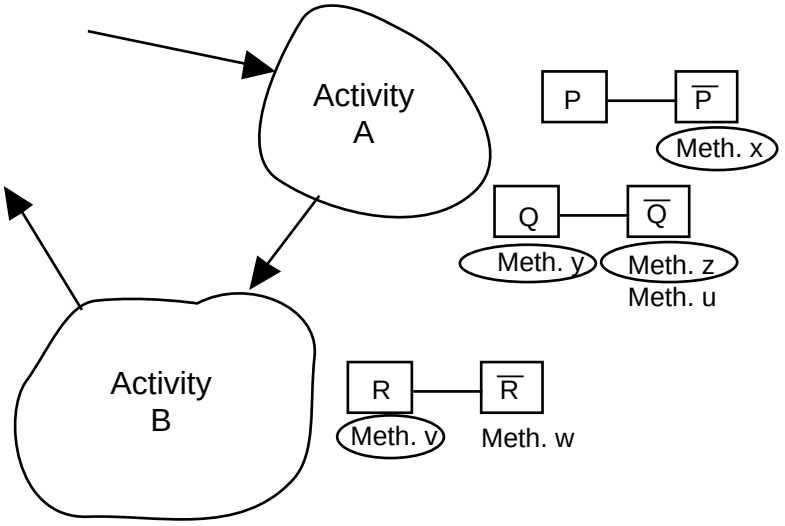


Figure 8.5: *Deciding which methodologies or parts hereof to use: Establishment of a liaison.*

repertoire of systems developers is limited and choice of methodology (i.e. establishment of liaison) is guided by the limitations of the repertoire and how it is viewed by the actors involved. Repertoires may be modified and changed, but it takes substantial experience, reflections, and adaption to the specific organisation, cf. Lesson 27.

Second, Lesson 19 stated that it is better to combine methodologies based on different Weltanschauungen than using a single methodology. In terms of liaison, this means that it is useful to design a liaison that takes more than one methodology, combine them, and relate them to the specific situation. (This is supported by several Lessons: 7, 8, and 18.)

Third, based on Lesson 21 it follows that the liaison should be designed such that technical and organisational Weltanschauungen, problem definition and problem solving, as well as individual and general standpoints are supported and balanced.

Fourth, from Lesson 22 it can be derived that a liaison does not at all need to be coherent. The parts of a liaison may co-exist or be in competition.

In summary, the notion of liaison is used to explicate a relationship between a specific situation and available methodologies, i.e. which methodologies to use and for what purpose. A liaison is thus the rationale

behind a choice of methodology. It is worth noticing that while we may point at useful combinations of methodologies it will often be an expression of more substantial or situational insight to explicate a useful liaison in stead. We may even gain from having flourish ways of expressing properties of liaisons, e.g. 'a useful liason', 'a powerful liason', 'a liason in contradiction', or 'a dangerous liaison.'

Activity 9: Adapt Methodologies to the Situation at Hand

This is the activity where the consequences of the decision in Activity 8 is taken. The established liaison is implemented.

This activity introduces the notion of *adaptation* of the methodologies of the liaison to the specific situation. The liason tells 'which' methodologies to use and 'why', i.e. a rationale in terms of some systemic argumentation and some methodlogical reasoning. A liaison together with an adaptation is the thinking behind a specific working practice and is in this sense very much like a methodology itself.[7] Here it is very simple, namely, an adaptation brings together the two aspects:

- 'What' is each of the methodologies going to be used for.
- 'How' is each of the methodologies going to be used?

By having introduced the notion of adaptation, it is now possible to think about and debate explicit suggestions for an adaptation. The relationship between methodologies, on the one hand, and a specific situation, on the other hand, can consequently be handled explicitly.

The simple way to adapt a liaison is by taking the models from Activity 8 and ask the following questions for each of the elements of the decision: (i) what part of the methodology is to be used? (ii) what further skills are needed? (iii) how is this obtainable? (iv) how is it to be used? (v) how is efficacy to be measured?

Activity 10: Use Methodologies

Not very much can be said about this activity. The methodologies have to be used after having been adapted during Activity 9. Knowing that Activity 6 depends on this activity it is worth considering how this dependency can be established in a specific situation. Monitoring how this activity is performed is a feasible way, e.g. by diary writing.

[7] It is worth noticing that a *adaptation* is referred to by the Lancaster School as a methodology, cf. (Atkinson 1986) and the discussion in Section 2.3.

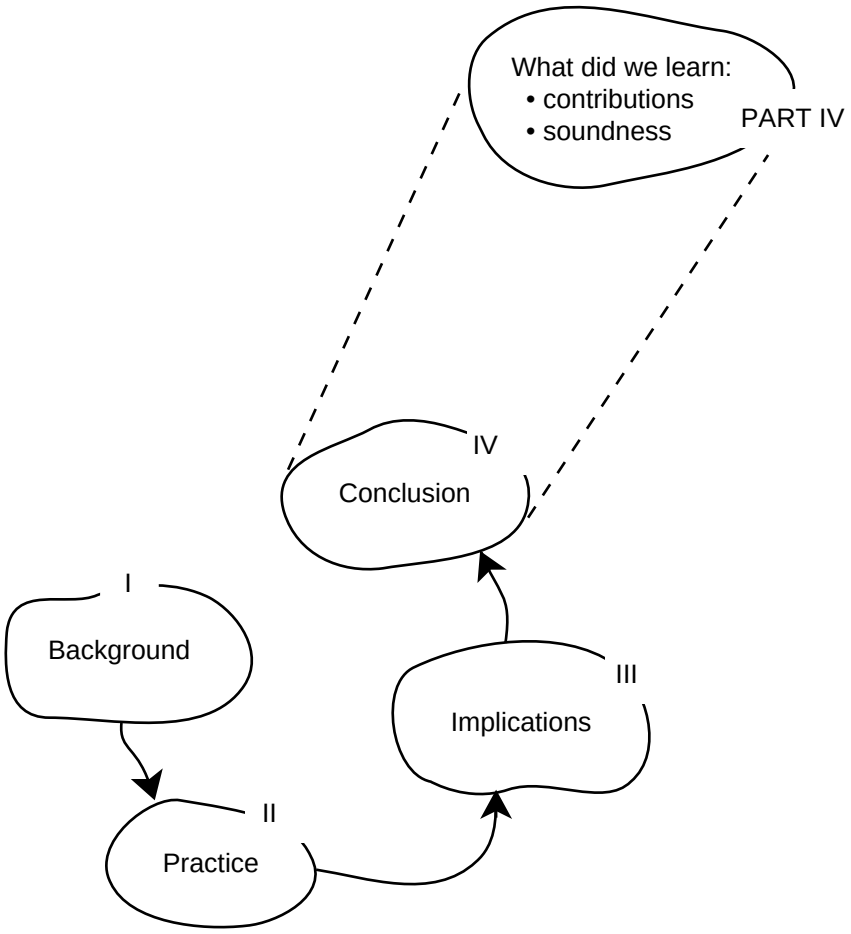
8.3 Summary

A new approach for learning and using methodologies in information systems analysis and design has been outlined. Intellectually, it is a specialisation of SSM and it borrows all its ideas about learning and problem solving from this general methodology.

It is an approach based on the lessons established by using and learning methodologies at three levels of interest, Part 2. Thus, it utilises the following main ideas:

- The criteria for using methodologies, cf. Chapter 7.
- The close relationship between using and learning.
- The importance of 'limited repertoire'.
- The criteria for combining methodologies.
- The notion of liaison.

Part IV
Conclusion


What did we learn:
• contributions
• soundness
PART IV
IV
Conclusion
I
Background
III
Implications
II
Practice

9

Thesis Conclusions

This chapter concludes the thesis. Section 9.1 summarises the outcome of the research that has been reported in this thesis as to bring about an understanding of the contributions to the field of learning and using information systems methodologies. Section 9.2 discusses the research approach in terms of the soundness of the lessons learned and the conclusions drawn.

9.1 Contributions to the Field

The outcome of the research is the lessons of Part 2 and the generalisations of Part 3. The lessons relate to three levels of methodological reasoning while the generalisations are formulated as a set of criteria for using methodologies and a new approach for learning methodologies.

The Lessons

At the individual level it was found that there is a rich variety, a diversity, of views held by systems developers on the use of methodologies. The diversity can to some extent be understood as an expression of individual differences of experience and attitudes towards professionalism associated with the systems developers. These lessons provide understanding of the contradiction between relying on methodologies and relying on experience and background.

At the project level it was found that the three methodologies and the theory (SA/SD, JSD, MOW, and TC) all relate to different domains of usefulness. This gave rise to several lessons about the Weltanschauung-

en, modelling languages, and frames of action of these four approaches. The four approaches were used in three different ways: commitment to a single methodology, relying on an organisational theory, and combining two methodologies. This gave rise to a number of lessons about these three ways of using approaches. The limitations of the lessons learned at the project level stems from the fact that not all differences in working practice are due to the use of methodologies, cf. Lesson 23.

At the organisational level it was suggested to base the introduction and learning of methodologies on argumentation, the interaction between theoretical and practical activities, and advice from methodology staff. A decision about which methodologies to use should be based on an evaluation of existing repertoires, on needs for development, and on feasible strategies for learning. The established methodological traditions in an organisation was identified as one of the key factors in carrying out these activities.

The Generalisations

The generalisations that are based on the lessons of Part 2 have been formulated as a set of criteria and an attempt towards a new approach.

The Use-Criteria are based on the lessons learned with SA/SD, JSD, MOW, TC, and to some extent with SSM. Each criterion has been formulated as a simple distinction between two aspects, and emphasises a choice to be taken. The criteria provide a framework where attention is drawn to the three main themes: domain of use (α), conditions for use (β), and characteristics of use (γ). In this way the methodologies have been related to situations where they are useful.

The New Approach is an approach based on the ideas of SSM and it utilises the Use-Criteria and the notion of limited repertoire in learning about which methodologies to use in a particular situation.

Comparing with Other Approaches

Chapter 3 provided a model that may now be used for relating the outcome of the research to other approaches for learning and using methodologies, appreciation approaches for short. Figure 9.1 shows the conceptual model where we for each activity may ask the following two questions. The first question: How was it done and what was achieved by this? The second question: What are the guidelines as to how it should be done and what can be achieved by this? The first question relates to what has been done as part of the research and the second question relates to the use of the approach by systems developers.

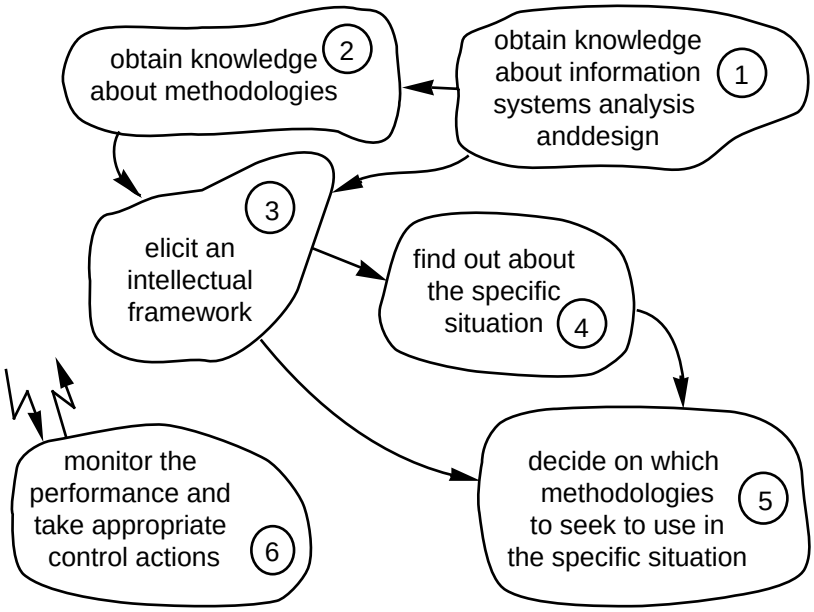


Figure 9.1: *The conceptual model used in the survey in Chapter 3. Key:* ◯*: activity;* →*: dependency between two activities;* ⇝*: dependency between an activity and all other activities.*

The first question can be answered by looking at how the research was done. Knowledge about methodologies and information systems analysis and design has been obtained by Case 1, 2, and 3 and been formulated as a number of lessons (corresponding to Activity 1 and 2 in the model). An intellectual framework has been elicited by generalising the lessons into the Use-Criteria and the New Approach (corresponding to Activity 3). The remaining activities of the comparison model have not been performed as part of developing the approach, but Case 3 gave insight into fundamental aspects of how they may be done. It is worth noticing that the outcome as a whole is a practice-based approach towards situations and methodologies, cf. Figure 3.3 on page 54.

The New Approach provides answers to the second question. Knowledge about methodologies and information systems analysis and design is obtained in the New Approach by activities 10, 6, and 7, i.e. by actually using the methodologies in a situation, by learning from their use, and by mapping the methodologies to the Use-Criteria, see Figure 9.2. The main effort in eliciting an intellectual framework has been done as part of the research, but it may be extended and modified in a small

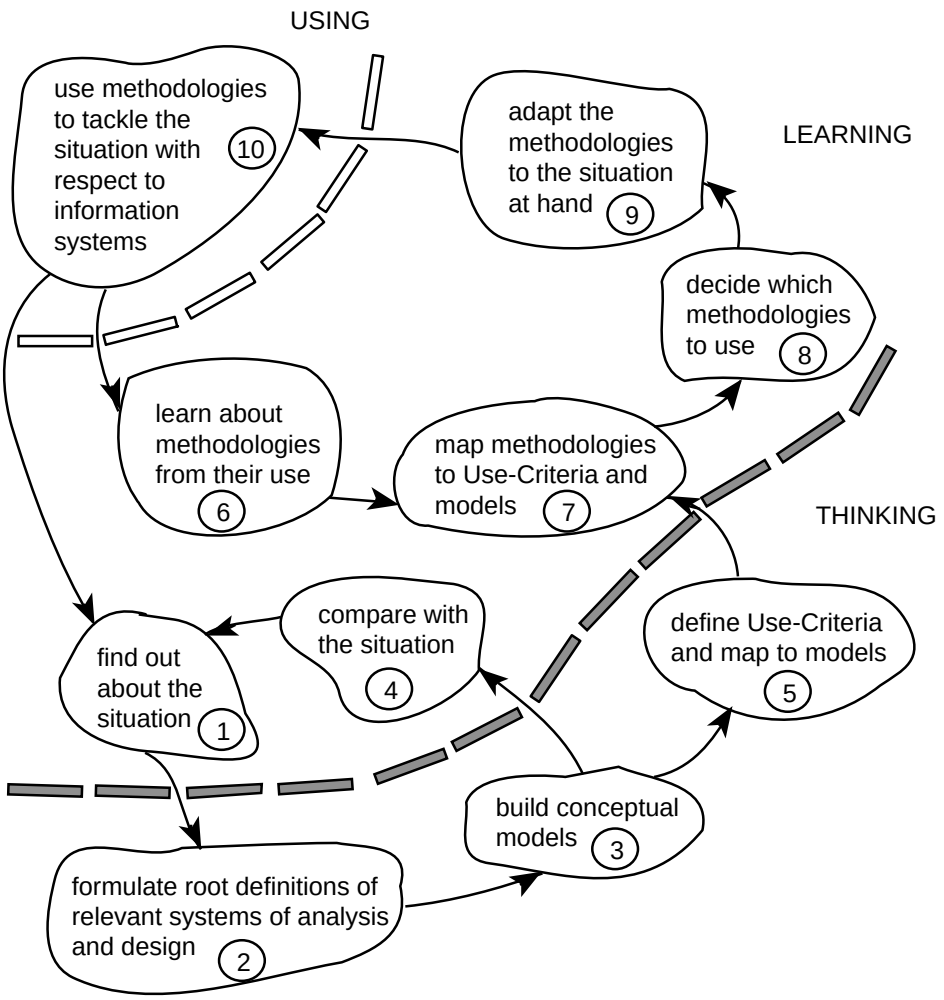


Figure 9.2: *The activities of the New Approach. Key: →: logical dependency; ◯: activity.*

scale by defining new and improved Use-Criteria (Activity 5), which will have consequences all the way through Activities 7, 8, 9 and 10. The finding out about the specific situation is done by the approach by carrying out Activities 1, 2, 3, and 4, i.e. the activities where systemic views are created and compared with the situation. The decision about which methodologies to use is taken as a result of Activities 5, 7, 8, and 9. In these activities, the systemic views are related to the criteria for using methodologies.

In continuation of the discussion in Section 3.2, it is fair to claim

that the New Approach is a contribution to a general approach in the sense that the Use-Criteria of Chapter 7 are general and offered to be applied in other situations. The New Approach is also a contribution to a situational and dynamic approach in the sense that it explicitly operates on unique systems views and a continued learning that eventually leads to new and improved criteria for using methodologies.

9.2 Soundness of The Research

Research of the kind reported in this thesis cannot be validated in the same way as a scientific experiment in a laboratory. For each conclusion we can, however, decide by ourselves whether an *argument* has been provided. Along the same lines as Weston I take an argument to be:

> a set of reasons or evidence in support of a conclusion.[1]

As a basis for deciding this I have throughout the thesis been explicit about the research approaches and the argumentation.

Three Research Approaches

I have applied three related but different research approaches. Each of these approaches has been described together with the actual research. The three approaches are simply different ways of doing action research.

Qualitative Interviewing was used at the individual level. The interviews with the systems developers did with relatively little effort provide substantial insight into the views held on the use of methodologies. At the same time, however, difficulties arise with this research approach because: (i) it is merely *views* that can be elicited, (ii) the process of interpreting the interviews is based on the researcher's selection of what is relevant and what is not, and (iii) the resulting lessons convey only fragments of what was actually said.

Diary Writing served as the means of documenting and reflecting on the working practice in each of the projects in Case 2. The strength of writing diaries for research purposes in projects is that they are valuable documents when examining the events and reflections afterwards. On the other hand, the limitations are that: (i) it takes a significant amount of resources to write the diaries while doing the projects, and (ii) there is little guidance on how to actually do the research, instead it is emphasised how to document it.

[1] (Weston 1987).

SSM was used as the methodology for intervention at the organisational level. Using SSM as a research approach is useful because it guides the research in the same way it guides any intervention, namely by explicating views-points and confronting these with the real world. The limitation of SSM as a research approach is that it is not as easy as with the two other approaches to provide context because the documentation is not as voluminous and rich.

Argumentation

I have all the way through this thesis strived at providing sound argumentation for lessons and conclusions. The following six aspects contribute to the soundness of the research and the report of it in this thesis.

- The assumptions and issues are explicit. Chapter 1 states the assumptions, research issues, and standpoints of the research as a whole. Each of the chapters in Part 2 starts by stating the research issues that are specific to the case reported. The three sections in Chapter 5 containing the three projects also state the issues relevant to these specific project.

- The lessons refer to practice. Reference to practice and experience is the prime vehicle for an argument.

- The lessons are given in context. All the lessons are preceded by a presentation of the context. This is done in Chapter 4 by presenting the three interviewees in their own words before going to the lessons that cut across the three persons. In Chapter 5 and 6 it is done in a description of the course of action in the practice referred to. Furthermore, context is given to the lessons where it is not sufficient with the general part.

- The lessons are given in depth and in detail. Each lesson is argued by describing and reasoning, sometimes at length, the path from the actual practice to the lesson.

- The generalisations refer to the lessons. The generalisations made in Part 3 are argued with explicit reference to the lessons of Part 2.

- The generalisations are made within an explicit theoretical framework. To the extent that the generalisations could not be made with specific reference to some of the lessons they were made based

on the ideas behind SSM. This framework is also used in interpreting the practice and the lessons.

It is now left to the reader of this thesis to decide whether I have provided sufficient reasoning and evidence to the conclusions I have reached.

A

The Lessons Learned

The Individual Level

Lesson 1 *Methodologies are never simply used. They are adapted to a particular situation and the guidelines of a methodology remain different from the working practice. (Page 71)*

Lesson 2 *There is significant variation in the degree to which different systems developers rely on methodologies. In one dimension it ranges from attempts to apply methodologies whenever possible to reluctance to use even the most straightforward ideas, tools and techniques of a methodology. In another dimension it ranges from attempts to apply methodologies literally to free interpretation of selected elements of a methodology. (Page 72)*

Lesson 3 *The way a systems developer uses methodologies reflects their experience and values. A systems developer with no or little experience is more likely to rely on methodologies than the experienced. A professional systems developer is more likely to find methodologies useful and necessary than the unprofessional. (Page 73)*

The Project Level

About SA/SD

Lesson 4 SA/SD *is based on the Weltanschauung that organisation and work can be seen as a computer system. Consequently,* SA/SD *is useful in*

domains characterised by routine activities. It is not suited for domains characterised by problem-solving. (Page 84)

Lesson 5 SA/SD *supports the development of a coherent set of reductionistic models of the information system viewed as a data processing system. The models can give an overview, but they do not provide insight into the information system as a whole. (Page 85)*

Lesson 6 SA/SD *sees systems developers as model builders and users as model checkers. The models can be understood by the users, but it requires analytical skills to build a model. (Page 86)*

About Using A Single Methodology

Lesson 7 *The idea of using a single methodology is not feasible. In a specific situation there will always be relevant matters that the methodology does not address. (Page 87)*

Lesson 8 *By using a methodology systems developers adopt a specific Weltanschauung. They tend to fit the real world into this Weltanschauung rather than being curious and critical. Furthermore, they impose the Weltanschauung on the communication with the users. (Page 88)*

About TC

Lesson 9 TC *is based on the Weltanschauung that organisation and work can be seen as transactions governed by contracts. It is a useful framework for understanding information needs of administrative organisations. (Page 94)*

Lesson 10 TC *invites the systems developers to build several models expressing different levels of detail and different contracts. (Page 94)*

Lesson 11 TC *is still insufficient as a frame of action. The concepts of* TC *are too imprecise and abstract, and the users do not understand the models sufficiently. (Page 95)*

About Using a Theory

Lesson 12 *A theory provides a Weltanschauung, parts of a domain-specific modelling language and an insufficient frame of action. Effective use of a theory requires not only analytical skills but also considerable resources for developing a working practice based on the theory. (Page 96)*

About JSD and MOW

Lesson 13 JSD *is based on the Weltanschauung that organisation and work can be seen as a computer system.* JSD *is useful in domains characterised by procedures and problem solving. But it requires that the domain is characterised by time-ordered events and absence of existing computer-based information systems. (Page 105)*

Lesson 14 JSD *supports modelling of the details of entities and functions together with their overall relationship. The models are useful for implementation considerations; but they are difficult to understand by the users. (Page 107)*

Lesson 15 MOW *is based on the Weltanschauung that organisation and work consists of a mixture of procedures and problem solving. It is useful as a general framework for understanding office work. (Page 109)*

Lesson 16 MOW *supports the modelling of offices from three different but related and useful standpoints. The users understand all the models and they are able to be constructive with models taking the standpoint of the individual. However, it requires analytical skills to build all the models. (Page 110)*

Lesson 17 JSD *and* MOW *support each other.* MOW *supports analysis based on an organisational Weltanschauung and* JSD *supports design based on a technical Weltanschauung. The insight gained by applying* MOW *is useful when applying* JSD *and vice versa. (Page 111)*

About Combining Methodologies

Lesson 18 *It can be efficacious and efficient to use combinations of methodologies in the same project. (Page 113)*

Methodologies at the Project Level

Lesson 19 *It is better to use a combination of methodologies based on different Weltanschauungen than to use a single methodology. (Page 114)*

Lesson 20 *A fully operational methodology is not sufficiently flexible and a theory is not sufficiently precise. A useful balance can be achieved by combination. (Page 115)*

Lesson 21 *Combinations of methodologies should support and balance: (i) technical and organisational Weltanschauungen, (ii) problem definition and problem solving, and (iii) individual and general standpoints. (Page 115)*

Lesson 22 *The Weltanschauungen in a combination may interact by co-existence or competition. (Page 117)*

Lesson 23 *Differences in outcome of a project is not only due to methodological differences, they are also due to other aspects like differences in the systems developers' proficiency in adapting methodologies. (Page 117)*

The Organisational Level

Learning of New Methodologies

Lesson 24 *Successful introduction of new methodologies requires relevant reasoning. The arguments need to be relevant to systems developers, management, and users. The arguments need to be in terms of the organisation's methodological strategies and purposes and in accordance with experiences in using other methodologies. (Page 130)*

Lesson 25 *Learning new methodologies is an interaction between theoretical and practical activities. The theoretical activities involve attending courses and discussing and reflecting upon the usefulness of the methodologies. The practical activities involve experiments and use of the methodologies in order to learn from mistakes and get experience. The interaction takes considerable time and other kinds of resources. (Page 134)*

Lesson 26 *In the introduction and learning of methodologies it is useful to have methodological advisors supporting and supervising systems developers. For each piece of methodological advice it is useful for the methodological advisors to understand the organisational support associated and how it contributes to quality and efficiency. (Page 135)*

Methodological Choice

Lesson 27 *The methodological repertoire of systems developers is limited. A repertoire can be developed, but it is necessary to build up substantial experience to make it useful. A repertoire needs to be continuously modified and adapted to the systems developers' and the organisations' needs. (Page 137)*

Lesson 28 *A choice of methodology in a particular situation is guided more by repertoire and personal preferences than by what, from a rational view-point, seems to be optimal. (Page 139)*

Lesson 29 *The process of choice of appropriate and adequate methodologies is a learning process related to a specific organisation involving both practical use of methodologies and reflections upon the use. (Page 139)*

Bibliography

(Andersen *et al.* 1984) Niels Erik Andersen, Vibeke Borum Bech, Elise Bitz, Lars Mathiassen, Preben Pilegaard, and Pål Sørgaard. Systemudvikling i praksis: Sparekassernes datacenter. MARS-report 4, Dept. of Computer Science, Aarhus University, 1984.

(Andersen *et al.* 1990) N.E. Andersen, F. Kensing, M. Lassen, J. Lundin, L. Mathiassen, A. Munk-Madsen, and P. Sørgaard. *Professional Systems Development.* Prentice-Hall, Englewoos Cliffs, N. J., 1990.

(Atkinson 1986) C. J. Atkinson. Towards a plurality of Soft Systems Methodology. *Journal of Applied Systems Analysis*, 13:19–31, 1986.

(Avison and Fitzgerald 1988) D. E. Avison and G. Fitzgerald. *Information Systems Development: Methodologies, Techniques and Tools.* Blackwell Scientific Publications, Oxford, 1988.

(Bengtson *et al.* 1984a) S. Bengtson, P. Huber, and P. A. Nielsen. Newmans kontormodeller (Newman's models of office work). Dept. of Computer Science, Aarhus University, 1984.

(Bengtson *et al.* 1984b) S. Bengtson, P. Huber, and P. A. Nielsen. Søndre dagbog (Project diary, South branch). Dept. of Computer Science, Aarhus University, 1984.

(Bengtson *et al.* 1985a) S. Bengtson, P. Huber, and O. Jensen. Analyse og design på kontorområdet (Analysis and design in offices). Technical Report IR-60, Dept. of Computer Science, Aarhus University, 1985.

(Bengtson *et al.* 1985b) S. Bengtson, P. Huber, and P. A. Nielsen. Vurdering af fremgangsmåden, Søndre afdeling, (Evaluation of the approach, South Branch). Technical Report KANDIS 6, Dept. of Computer Science, Aarhus University, 1985.

(Benyon and Skidmore 1987) D. Benyon and S. Skidmore. Towards a tool kit for the systems analyst. *The Computer Journal*, 30(1):2–7, 1987.

(Bergland 1986) G. D. Bergland. A guided tour of program design methodologies. *Computer*, pages 13–37, October 1986.

(Bjerknes *et al.* 1987) Gro Bjerknes, Pelle Ehn, and Morten Kyng, editors. *Computers and Democracy*, Aldershot, 1987. Avebury/Gower.

(Bødker *et al.* 1987) S. Bødker, P. Ehn, J. Kammersgaard, M. Kyng, and Y. Sundblad. A utopian experience: On design of powerful computer-based tools for skilled graphic workers. In Bjerknes et al. (1987), pages 251–278.

(Brandt 1983) I. Brandt. A comparative study of information systems design methodologies. In Olle et al. (1983).

(Budde *et al.* 1984) R. Budde, K. Kuhlenkamp, L. Mathiassen, and H. Züllighoven, editors. *Approaches to Prototyping*. Springer Verlag, Berlin, 1984.

(Checkland and Wilson 1980) P. B. Checkland and B. Wilson. Primary task and issue-based root definitions in systems studies. *Journal of Applied Systems Analysis*, 7, 1980.

(Checkland 1981) P. B. Checkland. *Systems Thinking, Systems Practice*. wiley, 1981.

(Checkland 1985a) P. B. Checkland. Achieving "desirable and feasible" change: An application of Soft Systems Methodology. *Journal of Operational Research Society*, 36(9):821–831, 1985.

(Checkland 1985b) P. B. Checkland. From optimizing to learning: A development of systems thinking for the 1990's. *Journal of Operational Research Society*, 36(9):757–767, 1985.

(Chen 1976) P. P. Chen. The entity/relationship model – towards a unified view of data. *ACM Transactions on Database Systems*, 1(2), 1976.

(Ciborra *et al.* 1980) C. Ciborra, G. Bracchi, and P. Maggiolini. A multiple-contingency review of systems analysis methods and models. In Lucas et al. (1980), pages 47–60.

(Ciborra 1981) C. U. Ciborra. Information systems and transactions architecture. *International Journal of Policy Analysis and Information Systems*, 5(4):305–324, 1981.

(Coad and Yourdon 1990) P. Coad and E. Yourdon. *Object-Oriented Analysis*. Yourdon Press, Prentice-Hall, Englewoos Cliffs, N. J., 1990.

(Davis 1982) G. B. Davis. Strategies for information requirements determination. *IBM Systems Journal*, 21(1):4–30, 1982.

(Davis 1988) A. M. Davis. A comparison of techniques for the specification of external systems behavior. *Communications of the ACM*, 31(9):1098–1115, 1988.

(DeMarco 1978) T. DeMarco. *Structural Analysis and Systems Specification*. Yourdon Inc., New York, 1978.

(Ehn and Kyng 1987) P. Ehn and M. Kyng. The collective resource approach to systems design. In Bjerknes et al. (1987), pages 17–58.

(Ehn and Sandberg 1979) P. Ehn and Å. Sandberg. God utredning (Good enquiry). In Å. Sandberg, editor, *Utredning och förändring i förvaltning*. Liber, Stockholm, 1979.

(Ehn 1988) P. Ehn. *Work-Oriented Design of Computer Artifacts*. Arbetslivscentrum, Stockholm, 1988.

(Episkopou and Wood-Harper 1986) D.M. Episkopou and A.T. Wood-Harper. Towards a framework to choose appropriate IS approach. *The Computer Journal*, 29(3):222–228, 1986.

(Flavin 1981) M. Flavin. *Fundamental Concepts of Information Modelling*. Yourdon Press, New York, 1981.

(Floyd *et al.* 1989) C. Floyd, F.-M. Reisin, and G. Scmidth. STEPS to software development with users. In *ESEC 1989, 11th - 15th Sept.*, University of Warwick, 1989.

(Floyd 1984) Christiane Floyd. A comparative evaluation of system development methods. Seminar paper, Technische Universität Berlin, Institut für Angewandte Informatik, Berlin, March 1984. Seminar at Dansk Datamatik Center, Denmark.

(Floyd 1986) C. Floyd. A comparative evaluation of system development methods. In Olle et al. (1986).

(Gould 1971) I. H. Gould, editor. *IFIP Guide to Concepts and Terms in Data Processing.* North-Holland, Amsterdam, 1971.

(Jackson 1983) M. Jackson. *Systems Development.* Computer Science. Prentice Hall, London, 1983.

(Jayaratna 1986) N. Jayaratna. Normative information model-based systems analysis and design (NIMSAD): A framework for understanding and evaluating methodologies. *Journal of Applied Systems Analysis*, 13:73–88, 1986.

(Jensen and Jepsen 1984) O. Jensen and L. O. Jepsen. Nordre dagbog (Project diary, North branch). Dept. of Computer Science, Aarhus University, 1984.

(Jensen and Jepsen 1985a) O. Jensen and L. O. Jepsen. Analyse og design inden for udlånsområdet, Vejlby afdeling, (Analysis and design of loan business, North branch). Technical Report KANDIS 3, Dept. of Computer Science, Aarhus University, 1985.

(Jensen and Jepsen 1985b) O. Jensen and L. O. Jepsen. Vurdering af fremgangsmåden, Vejlby afdeling, (Evaluation of the approach, North branch). Technical Report KANDIS 4, Dept. of Computer Science, Aarhus University, 1985.

(Jepsen and Nielsen 1986) L. O. Jepsen and P. A. Nielsen. Dagbogsskrivning til styring i systemudvikling (Diaries for effective management in information systems development). Master's thesis, Dept. of Computer Science, Aarhus University, 1986.

(Jepsen *et al.* 1989) L. O. Jepsen, L. Mathiassen, and P. A. Nielsen. Back to thinking mode — diaries as a medium for effective management of information systems development. *Behavior and Information Technology*, 8:207–217, 1989.

(Kling 1987) R. Kling. Computerization as an ongoing social and political process. In Bjerknes et al. (1987), pages 117–136.

(Land and Hirschheim 1983) F. F. Land and R. Hirschheim. Participative systems design: Rationale, tools and techniques. *Journal of Applied Systems Analysis*, 10:91–107, 1983.

(Land *et al.* 1980) F. Land, E. Mumford, and J. Hawgood. Training the systems analyst of the 1980's: Four analytical procedures to assist the design process. In Lucas et al. (1980).

(Lanzara 1982) G. F. Lanzara. The design process: Frames, metaphores, and games. In U. Briefs, C. Ciborra, and L. Schneider, editors, *Systems Design For, With, and By the Users*, pages 29–40. IFIP WG 9.1, North-Holland, Amsterdam, 1982.

(Lucas *et al.* 1980) H. C. Lucas, F. F. Land, T. Lincoln, and K. Supper, editors. *The Information Systems Environment.* North-Holland, Amsterdam, 1980.

(Lundeberg *et al.* 1979) M. Lundeberg, G. Goldkuhl, and A. Nilsson. A systematic approach to information systems development, i: Introduction, ii: Problem and data oriented methodology. *Information Systems*, 4, 1979.

(Lyytinen and Lehtinen 1987) K. Lyytinen and E. Lehtinen. Seven mortal sins of systems work. In P. Docherty et al., editors, *Systems Design for Human Development and Productivity: Participation and Beyond.* North-Holland, Amsterdam, 1987.

(Lyytinen 1986) K. Lyytinen. *Information Systems Development as Social Action: Framework and Critical Implications.* PhD thesis, Department of Computer Science, University of Jyväskylä, 1986.

(Mathiassen and Munk-Madsen 1985) Lars Mathiassen and Andreas Munk-Madsen. Formalizations in systems development. In H. Ehrig, C. Floyd, M. Nivat, and J. Thatcher, editors, *Formal Methods and Software Development.* TAPSOFT, Springer Verlag, Berlin, 1985.

(Mathiassen and Nielsen 1989) L. Mathiassen and P. A. Nielsen. Soft systems and hard contradictions: Approaching the reality of information systems in organisations. *Journal of Applied Systems Analysis*, 16:75–88, 1989.

(Mathiassen and others 1984) L. Mathiassen et al. Project contract. The Royal Case. Dept. of Computer Science, Aarhus University, 1984.

(Mathiassen 1981) Lars Mathiassen. *Systemudvikling og systemudviklingsmetode (Systems Development and Systems Development Method).* PhD thesis, Dept. of Computer Science, Aarhus University, 1981. DAIMI PB-136.

(Mumford 1987) Enid Mumford. Sociotechnical systems design: Evolving theory and practice. In Bjerknes et al. (1987), pages 59–76.

(Munkholm and Steffensen 1984) B. Munkholm and K. Steffensen. Østre dagbog (Pproject diary, East project). Dept. of Computer Science, Aarhus University, 1984.

(Munkholm and Steffensen 1985) B Munkholm and K. Steffensen. Vurdering af fremgangsmåden, Transaction Cost, (Evaluation of the approach, Transaction Cost). Technical Report KANDIS 2, Dept. of Computer Science, Aarhus University, 1985.

(Munkholm and Steffensen 1986) B. Munkholm and K. Steffensen. Reflektion over det transaktionsmæssige perspektive som grundlag for analyse og design på baggrund af et praktisk eksperiment inden for kontorområdet. Master's thesis, Dept. of Computer Science, Aarhus University, 1986.

(Munro and Davis 1977) M. C. Munro and G. B. Davis. Determining management information needs – a comparison of methods. *MIS Quarterly*, 1(2), 1977.

(Naur 1972) P. Naur. An experiment in program development. *BIT*, 12:347–365, 1972.

(Naur 1975) P. Naur. What happens during program development: An experiment. In M. Lundeberg et al., editors, *Systemering 75*. Studentlitteratur, Lund, 1975.

(Naur 1983) P. Naur. Program development studies based on diaries. In T. R. Green et al., editors, *Psychology of Computer Use*, pages 159–170. Academic Press, London, 1983.

(Newman 1983) W. Newman. Models for office work. Draft for chapters 4 and 5, not in final book, 1983.

(Newman 1986) W. Newman. *Developing Integrated Office Systems*. Information Systems. John Wiley, London, 1986.

(Nielsen 1984) P. A. Nielsen. Notat om dagbogsskrivning (A note on writing diaries). Dept. of Computer Science, Aarhus University, 1984.

(Nielsen 1987a) P. A. Nielsen. Projektforslag (Project charter, Provident case). Dept. of Mathematics and Computer Science, Aalborg University, September 1987.

(Nielsen 1987b) P. A. Nielsen. Reflections on development methods for information systems – A set of distinctions between methods. In P. Järvinen, editor, *The Report of the 10th IRIS Seminar*, pages 511–534. University of Tampere, 1987. Series B, Vol. 27.

(Nielsen 1988a) P. A . Nielsen. Oplæg til seminar om metoder, kvalitet og effektivitet i systemudvikling (Introduction to seminar on methodologies, quality, and efficiency in information systems development). Dept. of Mathematics and Computer Science, Aalborg University, February 1988.

(Nielsen 1988b) P. A. Nielsen. Feltnoter og protokol i projektperioden 22.9.1987 til 27.2.1988 (Field notes and protocol in the project period 22nd September 1987 to 27th February 1988). Dept. of Mathematics and Computer Science, Aalborg University. Personal Notes, 1988.

(Nielsen 1988c) P. A. Nielsen. Metoder til systemudvikling, konsulent rapport (Methodologies for information systems development, Consultancy report). Dept. of Mathematics and Computer Science, Aalborg University, May 1988.

(Nielsen 1988d) P. A. Nielsen. Oplæg til internt metodeseminar (Introduction to internal methodology seminar). Dept. of Mathematics and Computer Science, Aalborg University, February 1988.

(Nielsen 1989a) P. A. Nielsen. The emergence of contradictions in soft systems practice. In R. L. Flood, M. C. Jackson, and P. Keys, editors, *Systems Prospects: The Next Ten Years of Systems Research.* U.K. Systems Society, Plenum Press, New York, 1989.

(Nielsen 1989b) P. A. Nielsen. Reflections on information systems development methods: A set of distinctions. *Office, Technology and People*, 5(2):81–104, 1989.

(Nielsen 1989c) P. A. Nielsen. A survey of approaches in ISD methodology. In S. Bødker, editor, *Proceedings of the 12th IRIS.* Computer Science Department, Aarhus University, 1989. DAIMI PB-296.

(Nielsen 1990) P. A. Nielsen. Approaches for appreciating information systems methodologies: A soft systems survey. *Scandinavian Journal of Information Systems*, 2, 1990.

(Nygaard and Sørgaard 1987) K. Nygaard and P. Sørgaard. The perspective concept in informatics. In Bjerknes et al. (1987), pages 371–394.

(Olerup 1989) A. Olerup. Socio-technical design of computer-assisted work: A discussion of the ETICHS and Tavistock approaches. *Scandinavian Journal of Information Systems*, 1:45–73, 1989.

(Oliga 1988) J. C. Oliga. Mehodological foundations of systems methodologies. *Systems Practice*, 1(1):87–112, March 1988.

(Olle *et al.* 1982) T. W. Olle, H. G. Sol, and A. A. Verrijn-Stuart, editors. *Information Systems Design Methodologies: A Comparative Review.* IFIP, WG 8.1, North-Holland, Amsterdam, 1982.

(Olle *et al.* 1983) T. W. Olle, H. G. Sol, and C. J. Tully, editors. *Information Systems Design Methodologies: A Feature Analysis.* IFIP, WG 8.1, North-Holland, Amsterdam, 1983.

(Olle *et al.* 1986) T. W. Olle, H. G. Sol, and A. A. Verrijn-Stuart, editors. *Information Systems Design Methodologies: Improving the Practice.* IFIP, WG 8.1, North-Holland, Amsterdam, 1986.

(Ouchi 1980) W. G. Ouchi. Markets, bureaucracies and clans. *Administrative Science Quarterly*, 25, 1980.

(Patton 1980) M. Q. Patton. *Qualitative Evaulation Methods.* Sage, London, 1980.

(Pressman 1977) R. S. Pressman. *Software Engineering. A Practitioner's Approach.* McGraw-Hill, New York, 1977.

(Rapoport 1970) R. N. Rapoport. Three dilemmas in action research. *Human Relations*, 23(6):499–513, 1970.

(Sage *et al.* 1983) P. Sage, B. Galling, and A. Lagomasino. Methodologies for determination of information requirements for decision support systems. *Large Scale Systems*, 5:131–167, 1983.

(Schön 1983) Donald Schön. *The Reflective Practitioner - How Professionals Think in Action.* Basic Books, New York, 1983.

(Shomenta *et al.* 1983) J. Shomenta, G. Kamp, B. Hanson, and B. Simpson. The application approach worksheet: An evaluation tool for matching new development methods with appropriate applications. *MIS Quarterly*, pages 1–10, December 1983.

(Sirbu and others 1984) M. Sirbu et al. OAM: An office analysis methodology. *Behavior and Information Technology*, 3(1):25–39, 1984.

(SSADM 1984) Learmonth & Burchett Management Systems, London. *SSADM Training Manual*, December 1984.

(Stage 1989) J. Stage. *Mellem tradition og nyskabelse — Analyse og design i systemudvikling (Between Tradition and Creation — Analysis and Design in Systems Development).* PhD thesis, Dept. of Informatics, Oslo University, 1989.

(Taggart and Tharp 1977) W. Taggart, Jr. and M. O. Tharp. A survey of information requirements analysis techniques. *Computing Survey*, 9(4):273–290, 1977.

(Warmington 1980) A. Warmington. Action research: Its methods and its implications. *Journal of Applied Systems Analysis*, 7, 1980.

(Wasserman *et al.* 1983) A. I. Wasserman, P. Freeman, and M. Porcella. Characteristics of software development methodologies. In Olle et al. (1983).

(Weston 1987) A. Weston. *A Rulebook for Arguments.* Avatar Books, Cambridge, 1987.

(Williamson 1979) O. Williamson. Transaction cost economics: The governance of contractual relations. *The Journal of Law and Economics*, 22:233–261, 1979.

(Wilson 1984) B. Wilson. *Systems: Concepts, Methodologies, and Applications.* John Wiley, Chichester, 1984.

(Wood-Harper and Fitzgerald 1982) A. T. Wood-Harper and G. Fitzgerald. A taxonomy of current approaches to systems analysis. *The Computer Journal*, 25(1):12–16, 1982.

(Yadav *et al.* 1988) Surya B. Yadav, Ralph R. Bravoco, Akemi T. Chatfield, and T. M. Rajkumar. Comparison of analysis techniques for information requirement determination. *Communications of the ACM*, 31(9):1090–1097, 1988.

(Yourdon and Constantine 1979) E. Yourdon and L. L. Constantine. *Structured Design: Fundamentals of A Discipline of Computer Program and Systems Design.* Yourdon Inc., New York, 1979.

(Yourdon 1982) Edward Yourdon. *Managing the Systems Life Cycle.* Yourdon Inc., New York, 1982.