

Elements of a Theory of Design Artefacts

a contribution to critical systems development research

Olav W. Bertelsen

Ph. D. Thesis

Department of Information and Media Science

Aarhus University

Aarhus, January 1998

Summary

The present thesis offers a materialist and dialectical framework for understanding and influencing the field of use and design of computer artefacts. The framework is materialist because it insists on the reality of the material world and the material character of mental phenomena; it is dialectical because it rejects the idea that human life is a mechanical product of its material basis; the mental and the material are mutually determining in a dialectical relation. The thesis points to activity theory as a possible basic vocabulary in systems development research; integrating the relevant and necessary aspects involved in designing computer artefacts, not as being operational at a detailed technical level, but as a general “world view” under which relevant sub-fields are integrated.

The thesis emphasises material mediation in design by introducing the concept of design artefacts as a unifying perspective on systems development. This concept is based on a dialectical materialist approach comprising activity theory as a general perspective (mainly Engeström), and specifically the notion of primary, secondary and tertiary artefacts (Wartofsky), this background is complemented with the notion of boundary objects (Star), as mediators in boundary zones. The argument of the thesis is based on the tenet of activity theory that human praxis is mediated by artefacts and is continually changing in the process of sociocultural development; and the notion of historical crystallisation of praxis into artefacts. Systems development is understood as a zone where heterogeneous praxes meet to change a given praxis through the construction and introduction of new (computer) artefacts; this zone is mediated by design artefacts, which make different sense to the various praxes (boundary objects). Based on Wartofsky’s vocabulary, the thesis argues that design artefacts are clusters of primary, secondary and tertiary artefacts, each class simultaneously mediating different elements of the design process. As special instances of design artefacts transformers and abductors are discussed. Transformers are introduced as the artefacts mediating design as a transformation process emulating the process of development of artefacts in use. Abductors, as a class of design artefacts mediating the development of radically new motives, are briefly proposed.

Four main themes are addressed by the thesis: Firstly, the notion of design artefacts as an integrating perspective on systems development research and praxis, is introduced and developed. Secondly, a uniform notion of development tying use and design together, is discussed in relation to designing for development in use, and in relation to the notion of design as the trans-

formation of artefacts. Thirdly, a radically pragmatic philosophy of science based on the understanding of theories as design artefacts, is proposed. Finally, the issue of tertiary artefacts as mediators of innovation and creativity, setting an agenda for a dialectical materialist theory not neglecting the individual genius, is programmatically pointed to. This last issue is related to the changing concept of development in activity theory.

Resumé (Danish summary)

Den foreliggende afhandling, fremlægger et dialektisk materialistisk begrebsapparat for forståelse og handling i forhold til design af computer artefakter. Begrebsapparatet er materialistisk, fordi der insisteres på den materielle verdens realitet og på den materielle karakter af mentale fænomener; dialektisk fordi det afviser forestillingen om at menneskers livsudfoldelser er mekanisk determineret af disses materielle grundlag; det mentale og det materielle er gensidigt determinerende i en dialektisk relation. Afhandlingen peger på virksomhedsteorien som et muligt sæt af grundbegreber, der integrerer relevante og nødvendige aspekter af design af computer artefakter. Disse er ikke nødvendigvis operationelle i forhold til detaljerede tekniske og matematiske aspekter, men derimod udgør de et overordnet “verdensbillede”, under hvilket de forskellige underdiscipliner og disses teorier kan integreres.

Den materielle mediering af design betones gennem introduktionen af begrebet design artefakter som et samlende begreb i systemudviklingsforskning og -praksis. Det dialektisk materialistiske grundlag for dette begreb bygger generelt på virksomhedsteorien (især Engeström) og mere specifikt på begreberne primære, sekundære og tertiære artefakter (Wartofsky). Dette fundament kompletteres med begrebet boundary objects (Star) som mediatorer i grænse-zoner. Argumentationen er baseret på virksomhedsteoriens grundsætninger om at menneskelig praksis er medieret af artefakter, og at den konstant forandres i løbet af den sociokulturelle udvikling, samt at denne udvikling af praksis er krystalliseret i artefakter.

Afhandlingen fremstiller systemudvikling som en zone, hvor heterogene praksisser mødes for at forandre en given praksis gennem konstruktion og indførelse af nye (computer) artefakter. Denne zone medieres af design artefakter, der giver forskellig mening for de involverede praksisser (de er boundary objects). Baseret på Wartofskys begreber, argumenteres der for at design artefakter er klynger af primære, sekundære og tertiære artefakter, der hver især, men på samme tid, medierer forskellige elementer af designprocessen. Transformatorer og abduktorer nævnes som specifikke typer design artefakter. Transformatorer introduceres som den klasse af design artefakter, der medierer design forstået som en transformationsproces, der efterligner, eller fortsætter, den historiske udvikling af artefakter i brug som løbende krystallisering af virksomhed. Abduktorer antydes som en klasse af design artefakter, der medierer udviklingen af radikalt nye motiver.

Afhandlingen forholder sig til fire hovedtemaer: For det første introduceres og udvikles begrebet design artefakter som et integrerende perspektiv i systemudviklingsforskning og praksis. For det andet diskuteres et uniformt udviklingsbegreb, der binder brug og design sammen, dels i forhold til design, der støtter udvikling i brug, og dels i forhold til forståelsen af design som transformation af artefakter fra genstandsområdet. For det tredje foreslås en radikalt pragmatisk videnskabsteori baseret på forestillingen om, at teorier kan forstås som design artefakter. Endelig peges der programmatisk på tertiære artefakter som et centralt begreb i en dialektisk materialistisk teori om kreativitet og innovation, der ikke negligerer individets betydning. Dette sidste punkt relaterer sig til igangværende forandringer i virksomhedsteoriens begreb om udvikling og fremskridt.

Table of contents

Summary.....	i
Resumé (Danish summary).....	iii
Table of contents.....	v
Preface.....	vii
Overview of the thesis.....	ix
References to papers included in the thesis	xiii
PART I	
Introduction.....	1
Chapter 1: Two Empirical Cases	2
Debugger.....	3
Music Festival.....	8
Chapter 2: Systems development research	15
Chapter 3: A scientific basis?	19
Chapter 4: Human Activity.....	26
Heterogeneity and artefacts.....	40
Perception and action —classes of artefacts.....	42
Chapter 5: A notion of design and design artefacts.....	45
Design activity.....	45
A definition of design artefacts	49
Chapter 6: Exploring the notion of design artefacts	54
Theories as design artefacts	54
Principles versus praxis.....	57
Transformators the secondary artefactness.....	60
Abductors the tertiary artefactness.....	66
Conclusions of the thesis	69
References	75

PART II

First paper: An investigation into the design artefact concept as a vehicle for systems development research.....	85
Second paper: Fitts' Law as a Design Artefact: A Paradigm Case of Theory in Software Design.....	97
Third paper: Supporting the Development of Transparent Interaction	107
Fourth paper: Contradictions in the Festival Project: Activity systems, obstacles, and dynamic forces in design.....	123
Fifth paper: The Festival Checklist: design as the transformation of artefacts.....	141
Sixth paper: Organisational learning is crystallised into artefacts	160
Seventh paper: Understanding objects in use-oriented design.....	165
References to included papers	181

Preface

“All that is solid melts into air, all that is holy is profaned, and man is at last compelled to face with sober senses his real condition of life and his relations with his kind.” (Marx & Engels 1848).

The field of systems development research and praxis is haunted by a set of intertwined problems. Firstly, the accelerating development of basic technology and the constant rearrangement of societal relations imply that we cannot expect the field to develop as a craft. Secondly, the sociocultural constitution, in use, of meanings of computer artefacts implies that very few features of these are stable across time and context. Thirdly, the field tends to adopt any invention – technical, methodological, or conceptual – in a relativistically pragmatic way; thus, turning systems development into a mess of multi-paradigmatic babel. These fundamental problems point to the urgent need for a strong theoretical foundation for the integrated field of design and research. In order to be both practically viable and a proper guidance in this maelstrom, such a foundation has to be value-based, and it has to be both exclusive at the level of basic assumptions and inclusive in the operational assimilation of other perspectives.

The present thesis is my attempt to contribute to such a foundation for systems development research and praxis by introducing the dialectical materialist-based concept of design artefacts as a unifying perspective; by eliciting the relation between design and use in a uniform developmental perspective; by proposing a radically pragmatic philosophy of science; and by pointing to a materialist understanding of innovation and creativity.

A strong theoretical basis may seem to contradict with a radically pragmatic philosophy of science. However, I feel that it is necessary to reject both the positions treating scientific method as a universal guarantee and the relativist positions discarding the whole question of validity beyond immediate suitability. Even though we know the divorce statistics, we proclaim that we will love “till death”; and even though we know that no universal truth exists, we crave for truth in research. In the same way as it makes no sense to declare love until next Friday, it makes no sense to do research without striving for truth. A strong theoretical basis is largely a matter of taking the responsibility for the research we do, by ensuring that it makes sense beyond the most isolated context, and by caring for the sociocultural context of discovery and application.

Acknowledgements

This thesis is the result of a project carried out within the Devise Centre for Experimental Systems Development at Aarhus University, the project was funded by the Danish Research Programme for Informatics, grant number 5.26.18.19.

I will thank my supervisors, Kim Halskov Madsen, Susanne Bødker, and Morten Kyng. Kim has done a great job as my main supervisor at the Department of Information and Media Science, taking care of status reports and other administrative nuisances, as well as giving valuable comments on various parts of the thesis. Susanne has been a substantial source of inspiration and profound critique. Morten has been my father figure, always busy providing means of subsistence for the “family”.

The Devise group has been an exciting environment throughout the years, both as the base for realistic projects, and as a source of general inspiration, e.g. in the SubObj seminars, arranged by Morten Kyng and me, which were an important scene for developing a better understanding of the relation between object-oriented and user participation.

The Department of Computer Science has, in general, been a pleasant place to work at, in particular I want to thank Susanne Brøndberg and the rest of the secretarial staff who have been very professional and helpful in proof reading etc., often in chaotic situations.

I am thankful to the folks we worked together with in the music festival project, the designers I interviewed, and the people who participated in the Valhalla evaluation. I have benefited greatly from cooperation in specific projects with Jakob Bardram, Susanne Bødker, Henrik B. Christensen, Morten Kyng, Kim H. Madsen, Preben Mogensen. Also thanks to Preben for talking me into doing this thesis.

I owe thanks to The Danish National Centre for IT-research, for providing a supportive environment, and to the department of Information and Media Science for hosting my project.

Susanne Bødker, Kim Halskov Madsen, Preben Mogensen, and Finn Olesen have all contributed with valuable comments on various drafts of part one of the thesis. Also, a great thank to those not mentioned who in different ways have contributed to this thesis.

Most of all I will thank my family, Mette and Karl for their patient support.

Olav W. Bertelsen
Århus 1998

Overview of the thesis

The thesis is composed of a collection of published papers and a further elaboration of the introduced concepts tying the papers together. It consists of three parts: part one introduces the conceptual basis, elaborates further on the introduced ideas and sums up the conclusions; part two consists of the published papers.

Part one introduces and discusses the concept of design artefacts and outlines the theoretical and empirical prehistory for the concept. Chapter 1, “Two Empirical Cases”, describes the empirical background for the thesis. The section “Debugger” reports on a project that aimed at evaluating parts of the Mjølner BETA programming environment. The section “Music Festival” reports on a research project that aimed at developing computer support for the planning of a very large Danish music festival. That case has been the main background for the development of the notion of design as the transformation of artefacts. Chapter 2, “Systems development research”, discusses software and software design in general; critical systems development research (which the present thesis is a part of); and the basic features of system development research as opposed to the natural sciences. Chapter 3, “A scientific basis?”, introduces the notion of dialectical materialism in three steps: paraphrasing the development of the philosophy of science in the 20th century, discussing the role of basic assumptions in systems development research, and finally stating dialectical materialism through a discussion of Marx’ theses on Feuerbach. Chapter 4, “Human Activity”, outlines activity theory in very broad terms as a basis for a notion of design and design artefacts, together with Wartofsky’s analysis of the relation between perception and action, and Star’s notion of boundary objects. Chapter 5, “A notion of design and design artefacts”, introduces a notion of design activity and design artefacts based on the theoretical landscape outlined in the previous chapter. Design artefacts is the central concept developed in the thesis. Chapter 6, “Exploring the notion of design artefacts”, treats design artefacts in more detail; theories are discussed in terms of design artefacts both to illustrate the embracing power of that concept, and to get a better understanding of the relation between theory and design; the tension between principles and praxis is discussed as a basic feature of design artefacts; and the notions of transformers, transformation, and abductors are introduced. Part one is concluded with a chapter outlining the main achievements of the thesis, and possible directions for future work.

Part two consists of the seven publications included in the thesis. They are included in their original form, only the layout has been changed to improve the graphical appearance and readability of the thesis. Publication details can be found in the beginning of section two.

Paper one: An investigation into the design artefact concept as a vehicle for systems development research (1994a)

The main achievement of this paper is the tentative formulation of the design artefact concept. System development is discussed from the design artefact perspective based on the (at the time of the paper purely pragmatic) classification of design activities into *conception*, *communication* and *construction*, and the notion of secondary artefacts. It is suggested that the design artefact concept could be the basis for transcending the detrimental tendency to naturalism in object-oriented modelling; and the relation between theory and design is briefly discussed based on the introduced materialist pragmatism. An addendum to the paper points to the relation between tertiary artefacts and the experience of modernity.

Paper two: Fitts' Law as a Design Artefact: A Paradigm Case of Theory in Software Design (1994b)

With Fitts' law as an example, the paper unfolds the idea that theories can be understood and assessed as design artefacts. Three different roles (World view, tool, and metaphor) played by Fitts' law are identified based on examples from the literature. In turn, this insight is used in discussing the general relation between theory and design. The paper discusses and criticises the cognitive science based approaches to human-computer interaction research and praxis by questioning the value of additive models and the generalisability of laboratory data. The paper concludes that a radically pragmatic science of human-computer interaction taking the context of use, the context of design, and the relation between science and design into account can yield design-knowledge and understanding that goes beyond technical control; and that such a radical pragmatic science is necessarily based on dialectical, as opposed to mechanical materialism.

Paper three: Supporting the Development of Transparent Interaction (1995)

The paper, co-authored with Jakob Bardram, is an investigation into the concept of transparent interaction from the point of view of activity theory, defining transparency as handling the computer through operations. The focus of the paper is on how to support the user's development of operations (and thus transparent interaction) by embedding zones of proximal development in the artefact. The theoretical basis of the paper is Gal'perin's account on the formation of mental acts through the development of an orient-

ing basis, and the subsequent automatisisation of actions into operations, emphasising the quality of an action, in terms of its generality, its level of abbreviation and the extent to which the action is mastered. Three key issues in designing for transparency are identified: supporting development in use, ensuring a certain degree of initial familiarity and setting conditions for the formation of new operations. Finally, the paper points to the need for a thorough reconsideration of the theoretical foundation for human-computer interaction research and praxis, and points to activity theory as a possible solution.

Paper four: Contradictions in the Festival Project –Activity systems, obstacles, and dynamic forces in design (1996a)

In this paper, the Festival project is analysed by identifying contradictions in the course of the project, and then fitting these contradictions into the activity theory based notion of contradiction (Engeström, 1987). Thus, the paper is an investigation into operational use of activity theory in systems development. The primary contradiction between use and exchange value expressed as a contradiction in the festival of computers as utensils versus computers as epaulettes, is identified; which makes it possible to understand the main barriers in the project. It is concluded that the perspectives of contradiction and activity theory are valuable tools for making sense of design projects, and challenges for the further development of activity theory related to heterogeneity issues are identified.

Paper five: The Festival Checklist – design as the transformation of artefacts (1996b)

The paper describes the Festival project with focus on a “checklist”, and analyses the case in terms of the transformation concept introduced in the paper. The transformation concept is based on an understanding of crystallisation, secondary artefacts and boundary objects. The idea of transformation is that the ongoing process of crystallising evolving praxis into artefacts can be emulated in the use of representations during the design process. Transformation emphasises the materiality of praxis and aims at preserving praxis across the introduction of new technology. The strength and weakness of the transformation idea are that it limits radical innovation in ensuring that the new artefact makes sense to the considered praxis.

Paper six: Organisational learning is crystallised into artefacts (1996c)

Exemplified with the Festival case, the paper discusses the notion of organisational learning as the manifest crystallisation of collective experience into artefacts; thus, integrating an engineering and an emancipatory perspective within the framework of activity theory. The paper points to three issues in

design: crystallisation of procedural memory into new generations of artefacts; supporting continual crystallisation of organisational procedures and knowledge by facilitating tailoring and easy re-design; and disclosing current modus operandi through formalisation at the computer artefact, turning this into a Marxo-Freudian tool for building consciousness to transcend existing limitations through emancipating praxis.

Paper seven: Understanding objects in use-oriented design (1997a)

The paper is an investigation into the relation between participatory design and object-oriented methods. The concept of physical modelling is identified as a key principle in applying object-oriented methods in design with users, but it is also argued that many methods subscribe to a naive naturalism making object-orientation contradict with established understandings of human praxis. Based on Wartofsky's account on the role of representations and the general relation between perception and action, the role of physical modelling in design and in praxis is reformulated in non-naturalist terms. Based on the Festival case the transformation of artefacts concept is introduced as a possible framework for understanding the use of object-oriented methods in participatory design.

References to papers included in the thesis

- 1: Bertelsen, O. W. (1994a). An investigation into the design artefact concept as a vehicle for systems development research. In *Proceedings of the 17th Information systems Research seminar In Scandinavia IRIS 17, Syöte, Finland, August 1994*. pp. 715-125.
- 2: Bertelsen, O. W. (1994b). Fitts' Law as a Design Artefact: A Paradigm Case of Theory in Software Design. In Blumenthal, Gornostaev, & Unger (eds.). *Human-Computer Interaction. 4th International Conference, EWHCI '94 St. Petersburg, Russia, August 1994. Selected Papers*, Berlin: Springer Verlag. pp. 11-18.
- 3: Bardram, J. E. & O. W. Bertelsen (1995). Supporting the Development of Transparent Interaction. In Blumenthal, Gornostaev, & Unger (eds.): *Human-Computer Interaction. 5th International Conference, EWHCI '95 Moscow, Russia, July 1995. Selected Papers*. Berlin: Springer Verlag (LNCS 1015). pp. 79-90
- 4: Bertelsen, O. W. (1996a). Contradictions in the Festival Project - Activity systems, obstacles and dynamic forces in design. In Dahlbom, Ljungberg, Nuldén, Simon, Sørensen & Stage (eds.), *Proceedings of the 19th Information systems Research seminar In Scandinavia, (IRIS 19), 10-13 August 1996, at Lökeberg, Sweden*. pp.597-612.
- 5: Bertelsen, O. W. (1996b). The Festival Checklist: design as the transformation of artefacts. In Blomberg, J., Kensing, F. & Dykstra-Erickson (eds.). *PDC '96, Proceedings of the Participatory Design Conference*, Palo Alto: Computer Professionals for Social Responsibility. pp. 93-101.
- 6: Bertelsen, O. W. (1996c). Organisational learning is crystallised into artefacts [Position paper for workshop on organisational learning at CSCW'96]. In *SIGOIS Bulletin* vol. 17, no. 3, December 1996. pp. 37-39.
- 7: Bertelsen, O. W. (1997a). Understanding objects in use-oriented design, in Braa, Kristin & Monteiro, Eric (eds.). *Proceedings of the 20th Information systems Research seminar In Scandinavia, Oslo, 1997*. pp. 311-324.

PART I

Introduction

The present thesis introduces the concept of design artefacts as a unifying perspective in systems development research and praxis. Systems development, or design, is the activity in which new computer-based systems are brought about: it involves at least two parties – users and designers. The concept of design artefacts, in contrast to more process-oriented perspectives, emphasises material mediation. Part one gives a conceptual overview of the thesis based on the results reported in the papers contained in part two (Bertelsen 1994a, b, 1996a, b, c, 1997a, Bardram and Bertelsen 1995). The thesis is based on a dialectical materialist approach comprising activity theory (mainly Engeström 1987), and the notion of primary, secondary and tertiary artefacts (Wartofsky 1973), supplemented with the notion of boundary objects (Star 1989).

The argument is based on the tenet of activity theory that human praxis is mediated by artefacts and is continually changing in the process of socio cultural development. Development is not seen as an exceptional property of design, but as a basic feature of praxis in general, which in turn can be recognised in the artefacts mediating praxis. Thus, praxis is approached through mediating artefacts and, similarly, design can be comprehended by looking at design artefacts. Design is the zone where heterogeneous praxes meet to change a given praxis through the construction and introduction of a new artefact. This meeting is mediated by design artefacts which are boundary objects in the sense that they make different sense to the various praxes involved. Based on Wartofsky's analysis of the relation between perception and action, design artefacts are identified as being clusters of primary, secondary and tertiary artefacts, simultaneously mediating different elements of the design process. Transformers, mediating design as a transformation process emulating the process of development in use, and abductors mediating the development of new motives, are introduced as special instances of design artefacts.

The design artefact concept provides a uniform view on the outcomes of systems development research, (theories, methods, concepts, tools, etc.) as being mediators of design.

Chapter 1

Two Empirical Cases

This chapter describes empirical activities conducted throughout the project. These activities are directly and indirectly elaborated on in the papers, and in some cases also point to further writing and research.

During the autumn of 1993 Eval was started, as a joint effort of the Devise centre aiming at the evaluation, in an industrial setting, of the Devise environment for experimental systems development. Eval was motivated by a desire to see how the products of the Devise project would work in the “real world” context, as well as recommendations of the midway evaluation of the PIFT grant to Devise. For me, Eval was a promising source for empirical grounding of the study of design artefacts. Several companies were contacted, and in some cases project establishment activities were initiated. Promising activities were undertaken together with two departments of a Danish machine factory with whom we had several meetings where we planned the project and made initial field studies before the company backed out. The impossibility of finding an industrial partner for the evaluation may be ascribed both to reluctance in the industry to engage in projects without an immediately visible result, and to the state of the Devise tools at that time. The Eval effort ebbed away after approximately one year¹.

The Music Festival project which took place between autumn 1994 and autumn 1995, was in the outset planned to evaluate the Devise environment, but very early took another direction (see below).

In parallel with the Eval effort I conducted open ended qualitative interviews with two system developers in a small in-house development organisation, and with two designers of embedded software at a machine factory. The purpose was to acquire knowledge about how the creation of new ideas was mediated by design artefacts. Also in parallel with the Eval effort,

¹Later activities in which I did not participate, have more successfully contributed to the evaluation of the Devise environment. In the autumn of 1995 two student programmers were hired to implement versions of the Festival prototype and a hospital information system developed by Jakob Bardram, using the Devise environment. In 1997 the tools had reached a state where it was possible to apply them in an actual project. This was done in a project supported by CIT where people from the Devise centre cooperated with a big Danish shipping company on a new system.

Susanne Bødker and I conducted a project aiming at evaluating and giving input to redesign of the Mjølner debugger Valhalla.

The Valhalla evaluation and the Festival project are described in detail below. The description of the Valhalla evaluation presents the results of the evaluation, whereas the description of the Festival project is more of a raw case story.

Debugger

The aim of this project was to evaluate and give input to redesign of parts of the Mjølner BETA programming environment. The focus was on the object-oriented source level debugger Valhalla. The project consisted of two main activities, firstly an evaluation of Valhalla 1.3 focusing on observation of use, secondly contributing to the re-design of Valhalla through a design workshop with the developer in charge of the debugger and two other Mjølner/Beta designers. This section is based on a draft report written together with Susanne Bødker.

What is Valhalla

Valhalla is a source-level debugger for the BETA language and is a part of the Mjølner BETA System. The aim of Valhalla is to help the programmer locate errors in BETA programs by tracing their execution at the BETA source level. Thus, the Valhalla user often knows the code looked at and the purpose of the program rather well. And the debugger is launched because of an error in the program. There is at least one more reason for using Valhalla, namely to be able to look at a program execution in order to learn about the program. Thus, Valhalla is in some cases also of interest to persons who hardly know the code, and who are in a non-error situation. Valhalla includes the following functionality:

Controlling program execution: The program execution may be controlled e.g. by setting breakpoints and single stepping at the BETA source level. Runtime errors are caught by Valhalla which will display the offending object and code. From there, program state can be browsed to locate the cause of error. The debugged program executes in a process of its own, being watched by Valhalla, but otherwise unaffected.

Static browsing of program text: Static program browsing is supported through group- and code-viewers making it possible to browse the different relationships present in a BETA program. These include super-pattern relationships, links from name applications to the corresponding declarations,

the binding of slots to fragments, and so on. Finally, search for expressions denoting objects and patterns may be performed.

Dynamic browsing: Dynamic browsing refers to the possibility of browsing the dynamic state of the BETA program execution. This includes inspection of the call chain and object states whenever the program execution is stopped at a breakpoint or as a result of a runtime error.

The user interface of Valhalla consists of a main window containing a menu and a number of different windows (viewers) displaying different aspects of the debugged program.

Version 1.3 in use

The focus of the evaluation was on Valhalla's function in use, according to the general constitution of use quality in use, but also due to the practical difficulty of generating realistic debugging situations in the laboratory.

The study was done through the following activities: interview with the programmer in charge of the version of maintaining and developing Valhalla, as well as designing the next version. Interviews and demonstration with one frequent user of Valhalla. Our own novice attempts to explore Valhalla based on the user manuals. Study of a skilled Beta-programmer's introduction to Valhalla. Observations and video taping of a programmer's use of Valhalla. The last activity turned out to involve methodical problems.

Two interdependent methodical problems arose in relation to studying the debugger in use. The first problem was related to debugging being an integrated part of programming, happening at unforeseen occasions. Bugs are not isolated phenomena only related to the piece of code the programmer is working on when bugs occur. It is next to impossible to generate bug situations that in a realistic way can be used as a basis for laboratory testing; thus, we were thrown on hanging around at the programmer's office until bugs occurred. The method we applied was to have a video camera, that the programmer would switch on when using the debugger, installed at the programmer's workstation. This led to the second problem, related to making sense of the recorded data. The resolution on the video tapes was so low that the only thing we could distinguish on the recorded UNIX screen was the numerous windows popping up. The tapes partly worked for stimulated recall interviews, but we were unable to make micro level analysis.

Through the evaluation we were able to detect a number of problems in the use of Valhalla 1.3, the most important of which are briefly described below.

The number of windows. In Valhalla, any new undertaking is shown in a window: any object, stack etc. is shown in its own independent window. The number of windows launched is in itself big. Furthermore, several of the windows show snapshots, thus e.g. a new request for the stack later in the

session opens yet another stack window. It is later impossible to see which windows belong together, somehow representing the “status” of the program execution, or which version of e.g. the stack is the most recent, and which actually contains outdated information. The programmers typically do not want to keep the snapshots/state information between breaks in the program execution, but each window has to be closed individually.

Keeping track of history. Knowing which windows are created when and in which context is, as outlined above, one of the problems of keeping track of history. Furthermore, the programmers expressed a wish to be able to go backwards in the program execution, or to restart from a well-defined state.

Staying within one’s own code. Though Valhalla delimits which code may be looked at through the fragment system, there were three major problems with respect to which code one looks at. Firstly, Valhalla stops on signals that do not always come from errors in the code being debugged. Secondly, using “step into” takes the programmer through the execution of all the code including the basic libraries. What happens there is often of little interest to the programmer, and even worse: it is often exactly the step before getting outside ones own code that is interesting. This leaves the programmer in a difficult situation since there is no way of getting back. Thirdly, in various larger settings it is important to be able to debug only one’s own part of a larger application, e.g. when building something on top of the hypermedia, one does not want to debug the hypermedia.

Re-design workshop

To clarify the future of Valhalla, Susanne Bødker and I conducted a workshop with Søren Brandt (the developer responsible for the debugger), Elmer Sandvad and Jørgen Knudsen (designers at Mjølner). The workshop was divided into three phases. Firstly, a brief report on the preliminary findings from the evaluation, as described above. Secondly, a “Brain storm” centred around why, who, how, what questions of debugging. Thirdly, presentation of the new design by means of a paper mock-up constructed during the morning, and a simulated debugging session with the new version.

Brainstorm

The brainstorm was structured around the questions: Why? Who? How? and What? in relation to object-oriented source-level debugging. This structure was pragmatically chosen because of successful experience with the use of such questions, and was backed up theoretically by the three levels of human activity corresponding to the questions why? what? and how?

It turned out that the only participant using the debugger regularly was Søren. The two other participants were using more traditional techniques like inserting printouts of variables in the source code, which made it possi-

ble to trace critical variables, during the entire execution, particularly the initial phase, to detect the errors when they are generated. Their own explanation for not using Valhalla was lack of education, but it also pointed to the fact that Valhalla 1.3 did not support tracing of variables, but only views on “discrete” states of objects.

Valhalla is intended to be an *object-oriented* debugger in contrast to traditional source level debuggers with only one code window and one trace window and no support for monitoring the relation between different parts of the program. However, the essential thing to monitor in most debugging situations is dynamics. This discrepancy can be understood as an instance of the general tension between principles and praxis. The Mjølner BETA system is built upon a conceptual framework focusing on static aspects of modelling rather than dynamic aspects of execution. Thus, Valhalla 1.3 was constructed to conform to the principles, rather than to support programming praxis.

During the brainstorm it was questioned whether anything was gained from the object-oriented debugging approach, and it was agreed that the next version should support “traditional” source level debugging in addition to the object-oriented approach, e.g. by integrating a report generator or scripting of calculations relative to specified watch points. The advantage of this strategy was, according to Jørgen, that programmers could start using the tracing facilities and then gradually approach the new “religion”.

However, in dealing with totally unexpected errors no information can be obtained from traces, because you don't know what to trace. In such situations the object-oriented approach is more likely to help the programmer. Furthermore, Valhalla 1.3 had been a useful tool for reporting errors in basic libraries etc. Thus, the object-oriented approach is useful when dealing with different layers of a program simultaneously.

The principles versus praxis tension also turned up in the discussion of the difficulty of navigating between the excessive of windows in Valhalla 1.3. It was discussed whether programmers have, or should be allowed to have, a textual understanding of the program they are working on. Søren claimed that “if you are working on the text then you are working in a procedural manner” (as opposed to an object-oriented manner), but later on Jørgen claimed that “the program text is a suitable means of navigation”. It was stressed that “the object-oriented programmer” is not working on the code-text but on the structure of the program, that is hierarchies, patterns, objects. The view on the textual representation as well as the view on specific executions are only means for viewing the structure. Here, the principles of object-orientation prevented the Mjølner designers from acknowledging the central role of textual representations in programming praxis, including their own praxis. However, they agreed that a less fragmented (object-ori-

ented) and more sequential (textual) way of viewing the program would be fruitful.

Paper mock-up as a vehicle for discussion of the new design

Based on feedback from users and discussions at Mjølner Informatik, Søren had made a design specification for a new version of Valhalla, intending to solve most of the problems described above. In the new design all of Valhalla resides inside a single window. The main reason for this is that it makes it possible to draw connections between sub-window displaying objects, data or source code. The display of source code in the new Valhalla is done with a subset of the Mjølner hyper-structure editor Sif.

After the brainstorming the new design was presented by means of a paper mock-up, produced during the morning before the workshop. The mock-up was based on the specification for the new version and the standard example Beta program from the present Valhalla manual. Although Søren had a very clear vision of the new design, the process of building the mock-up provoked discussions and decisions about non-trivial details.

The presentation and enactment of the new debugger served as the vehicle for a long discussion about the new design and about how the different parts of the Mjølner Beta environment should be integrated in the future. Several times the discussion became very technical, but frequently the focus turned back to usability issues. No specific conclusions were drawn from these discussions; but they yielded new unstructured insights among the Mjølner designers into the environment they were building and the activity this environment was aimed at supporting.

Findings and further research

The most obvious problem with the studied version of Valhalla was the number of windows and the resulting risk of getting lost. This problem was easy to discover through isolated studies of the use of the debugger. What turned out during the workshop was that Valhalla mainly supports one aspect of debugging (the object-oriented view on runtime instantiations), and that this aspect wasn't enough debugging support in real programming. The textual aspects turned out to be equally important; both support for tracing and support for viewing the code.

In relation to my thesis, the study of the Mjølner BETA debugger elicited the contradiction between the object-oriented principles of working with structure and the actual text oriented programming praxis of both users and designers of the Mjølner BETA system. Thus, the study has contributed to understanding the tension between prescription and praxis, which is one of the important themes of the thesis.

A topic for further research is on methods for collecting data on programming, both at the technical level of being able to see what is going on, and at the level of finding a meaningful basic unit to study (debugging is obviously a too small unit). Further empirical studies of the object-oriented principle of structure and its role in practical programming, is another topic for further research. This will involve analysis on what is the object of programming work (text, structure, application, use praxis).

Music Festival

The Festival project took place during the first half of 1995, and involved Morten Kyng, Kim Halskov Madsen, Preben Mogensen, Henrik Bærbak Christensen, and Olav W. Bertelsen from the Devise Centre at Aarhus University, and people from the Festival organisation. During the fall of 1994 the Festival had decided that they needed an IT-strategy, and an internal IT group was formed. The project was initiated by the Festival who contacted the Devise Centre to initiate a project eliciting the possible advantages of introducing IT in the planning and production of the festival. Unfortunately, the scope of the project was cut down by festival management half way through, and the engagement terminated before the planned activities were completed. Nearly a year should go before any of us were able to write about it. Apart from the writings about the Festival case included in this thesis, Kim Halskov Madsen (1996) has used the case in a discussion of initiative in participatory design.

The Festival

The Festival is a non-profit organisation with the production of Denmark's largest annual music festival as its main objective. In 1995 the festival took place during 4 days, with concerts on 8 different stages, presenting a total of more than 140 different acts. Making a music festival involves many different tasks: engaging the artists, establishing camping areas for the audience, selling tickets, selling foods, controlling access to the festival site, informing the press, building the festival site, etc. The volunteers working in the Festival are organised in 35 operation groups, 150 are working throughout the year and in the time around the festival 2500 volunteers are enrolled. 9000 members of external organisations (e.g. boy scouts and sports clubs) are working during the festival. 10 people have a regular, paid job at the Festival.

The focus of the project was on technical production and pre-production, involving people from the Booking group, Sound and Light group, Transit

group, Catering group, and the eight stage groups (Green, Red, etc.). The Booking group is responsible for deciding which artists are going to play at the festival, and for negotiating the conditions and prices with the agents. The Sound and Light group is responsible for the technical side of the artists' performances. They are responsible for making arrangements with sub-contractors running the basic sound and light equipment on the individual stages, and for making sure that all the artists will have the conditions needed for their performances, equipment-wise, including instrument amplifiers, piano tuners, and help with special effects. In addition, the Sound and Light group has a co-ordinating role in the pre-production. The Transit group is responsible for the transportation of the artists from airport to hotels to and festival, etc., and for the booking of hotel rooms. The Catering group is responsible for dressing rooms, meals, and other backstage facilities for the artists. The festival takes place on eight different stages. The stage groups are responsible for the production at the stages including establishing the facilities in the backstage area, e.g. stage and production offices, stage hands, etc.

The project

The project took place during the first half of 1995. We had the first meeting with the Festival in the middle of January, and in the beginning of February we decided to engage in the project, and an agreement for the project was formed.

The first project meeting at the Festival office in Roskilde took place at the end of February. The Sound and Light group told us about the Festival from their perspective, and about their work. We in return demonstrated the Devise Hyper Media System as inspiration for the Sound and Light activists. The Sound and Light group had prepared for the meeting, by making two descriptions on paper, one describing the "flow of information" to and from the Sound and Light group during pre-production, the other a sketch of a database for pre-production represented as a screen layout.

During March a series of interviews with two of the stage groups, the Catering group, the group responsible for access to the festival area, the Transit group, the Booking group, festival management, and a secretary, were conducted. On April 1, a workshop with Sound and Light, Transit, Catering, and the Yellow stage groups took place. After the workshop we decided to use a database management system in order to have a prototype ready before the big rush of the pre-production activities. During April we designed and implemented a first prototype of a system for Pre-production.

At the middle of April, the Festival management became nervous about the project, fearing that too much information would flow too freely around in the organisation, therefore they dictated that the project could only continue

with the Sound and Light group. As a consequence the second and third planned workshops with the operation groups had to be cancelled. This breach of the original agreement made it difficult for us to continue the work in a decent manner, especially it became impossible to confront their understanding of the festival with the actual reality. Seen in retrospect we should have left the project at this time.

During May, the first version of the prototype was installed at the Sound and Light office, and we helped them entering some of the existing data into the system. This version was never used by the Sound and Light group. At the end of May a simplified version of the prototype more suited for the situation with Sound and Light as the only users, was installed. At this time, Sound and Light had not made the checklists as they did during the previous year's pre-production, some of the data were entered into the database, but most data were only available on faxes, and ad hoc notes. The final result was low quality checklists delivered to the stage groups.

In the last week of June, Festival 95 took place. During the festival we conducted field studies at the Festival site. After the festival we wrote a report on potential advantages of introducing IT in the Festival organisation. The first version of this report was sent to the festival on August, 25.

Due to time pressure we were not able to have the people we had observed or interviewed, approve the report. To remedy this violation of our own principles we both sent the report to all the involved groups as well as to the specific people who's addresses we knew, so they could correct possible mistakes. The reaction from festival management was quite surprising. In a letter dated September 5, they told us that we had violated the conditions for the project by sending the draft report to the volunteers. Management had the opinion that we had continued work without respecting the Festival's reality and the general reality, which according to management was that.

“a strategy must be ready before possible affected parties are involved. [if this had been respected] then the researchers would have avoided unrealistic expectations [among the volunteers] and subsequent frustration and disappointment” (Letter from the Festival management September 5, 1995, my translation).

Despite this letter and the general conditions for the project, we felt that several interesting topics were left to study in relation to the use of IT in the Festival, furthermore the Festival's internal IT group wanted to continue to cooperate with us. This lead us to engage in negotiations about a continued project. However, it was not possible to get sufficient explicit commitment from the Festival management, so we withdrew from further activities.

Pre-Production Work

During the spring, the head of the Sound and Light group is employed at the festival to take care of the technical pre-production, and to distribute relevant information to other operation groups. The most important means of communication throughout this process is telefax, and to some extent telephone. Pre-production work is a kind of detective work; when an artist is booked for the Festival the normal situation is that the only information the Festival gets is the name of an agency somewhere. Thus the first difficult task in pre-production is to find somebody who actually knows something about the artist, and then to convince this person that the festival needs up-to-date information as soon as possible. Pre-production work is complicated by several factors. People in the music business are always late with everything; it can be hard to make people understand that the festival needs information in advance. Also, it is very important for especially the bigger artists to show off by demanding specific resources for their appearance at the Festival, these demands then have to be negotiated in some way. Finally, information about the Festival program, and information about arrival times and hotels of the artists has to be treated confidentially. Program information has to be kept secret until it is given to the public to maintain the advertising value; information about hotels and arrival, to protect the artists during the booking negotiations and during the Festival (e.g. big stars do not like to have their hotels invaded by their fans and the press). The general understanding in the Sound and Light group is that the festival could be produced without pre-production, but that it then would be more chaotic. Thus the purpose of pre-production is to facilitate a smooth production with a relaxed and friendly atmosphere.

During pre-production Sound and Light builds a band file, a plastic folder enclosing documents, for each performance. These files are kept in a matrix of cardboard boxes, with one column per stage and one row per day. The first document in the file would normally be the checklist. One sheet of paper with the total plan of performances, the play plan, organised in the same way is used both as a tool for locating the files in the cardboard boxes, and for recording central information about the specific performances, e.g. the state of the information gathering, and the need for special equipment. The play plan is always situated on the desk in the Sound and Light office; when someone calls on the phone the Sound and Light person will look at the play plan, locate the artist in question, and examine the state of the pre-production for this performance; then the pre-production person will take the file in the cardboard box while continuing the discussion on the phone.

The checklist is a sheet of paper with pre-printed fields for information related to a specific artist's performance at the festival. The checklist was originally invented in the Green stage group. This list contained fields for all the information that should be available or collected when an artist arrived at the backstage area at the Festival site. The checklist was filled in when

the Sound and Light group handed information from the pre-production over to the stage groups, and it was later used when the artist arrived. Subsequently it was adopted by the Sound and Light group, and used during the pre-production process as the central overview of the individual artists. From 1993 the Sound and Light group produced a common checklist for all the stages, and filled in the available information about the artists before carrying the complete files over to the stage groups. Thus the checklist had three functions: as a tool for the collection of information during pre-production, secondly as a medium for forwarding information from the Sound and Light group to the stage groups, and finally as a tool at the stages when receiving the artists and carrying out the performances.

Designing for pre-production work

The checklist became the central point in the design process. Despite all the technological visions we had introduced to the users in the beginning of the project. The most important reason for that was that the Sound and light group already had a very strong vision about a relational database. One member of the Sound and Light group had earlier experiences with computer support for festival pre-production. This support was implemented in a relational database management system, and looked very much like a “smart checklist”. The Sound and Light group had discussed this concept and drafted a sketch of a relational database screen layout as they would like it. This database sketch was basically a slightly expanded transformation of the checklist.

According to the original plan, design together with the users should take place as a series of workshops; only one of these was realised. This workshop took place in the Festival buildings and was scheduled to 5 hours. The planned participants were members of four Festival operation groups: Sound and Light (3 persons), Yellow stage (one person), Catering (one person), and Transit (one person); the three seniors researchers and two apprentice participatory design researchers.

The plan for the workshop was to enact or simulate a series of work situations, both routine and problematic, from the planning (pre-production) and production of the festival. The participants were encouraged to bring real or made up situations that they found interesting, “focusing on the exchange of information” and how IT could be used, to the workshop. The idea was furthermore that we would introduce various kinds of technologies into the game to elicit how, e.g. computerised telefax, central and local databases, e-mail, or hypermedia would change work at the festival.

The workshop took place around a table, on the walls were mounted large pieces of paper. One piece of paper was laid out with columns for various kinds of technologies; local databases, centralised databases, hypermedia,

computer integrated telefax, etc. Cardboard lids were available to be used as database mock-ups, and yarn for simulating hyper-links between documents. Other pieces of wall paper were used to record situations and problems during the workshop. Material from the previous year's Festival was photocopied in advance together with some made-up ideal typical material produced by the Sound and Light group.

The first problem which we encountered at the workshop was that the participant from the Yellow stage never came, after an hour of waiting and several phone calls his seat was filled with one of the Sound and Light guys, who had previously worked in the Orange stage group. This resulted in a strong Sound and Light, and planning bias of the workshop; thus, and it became much harder to generate situations where the stage claimed not to have the information they needed. These situations would probably have arisen if the activist from Yellow stage had participated, because that group emphasised the lack of information during the preceding interviews.

The simulation games ended up focusing on how things were done the previous year; the workshop basically became a discussion repeating the information the researcher already got from the interviews. The cardboard lids and the yarn were never used, and the technology wall paper did not make its way into the situation. The design or construction related part of the workshop was limited to the last half hour, when the original database sketch, produced by the Sound and Light group, was examined with respect to suppliers and users of the information. This part of the workshop was important for building a prototype, but it did not break the meetingness of the workshop.

The design of the prototype took place right after the workshop. The first step was to make an object-oriented description of pre-production and production. The main functions of this description became to generate discussions between us about data formats, and to serve as a vehicle for the establishment of a shared understanding of the Festival among us in the Devise group. In this process the understanding of the Festival we got from the interviews was an important resource.

The transformation of the object-oriented description was done by mapping objects to tables in a straightforward manner. The issue of data-ownership distribution of the database over several non-networked PC's was already dealt with in the object-oriented model by reflecting the ownership of data in the division of objects. The construction of the user interface of the prototype started out on paper but we soon agreed that it was easier to program the interface right away without making a specification first. The task was uncomplicated because most of the prototype was specified in the Sound and Light database sketch, and on the pre-printed checklist made by Sound and Light the previous year.

Findings and further research

In relation to my thesis, the Festival project served as the main background in forming the notion of design as the transformation of artefacts (Bertelsen 1996b, 1997a). Furthermore, the project has been used as a testbed for Engeström's notion of contradiction as an analytical tool (Bertelsen 1996a), and the checklist transformation story has been used to illustrate the idea that organisational learning is crystallised into artefacts (Bertelsen 1996c).

The festival project gives rise to several topics for further research. A linguistic investigation into the object-oriented analysis and the database design could contribute to both understanding our work in the project and design work in general. The lacking ability of our methods to deal with use praxises going through long cycles of separated phases, calls for the development of new methods. Finally the issue of democracy in organisations with many volunteers, and how to understand such organisations, can yield insight into non-economical perspectives on the workplace.

Chapter 2

Systems development research

This chapter briefly sets the stage for the contribution of the thesis by discussing the basic constitution of software, and of software construction as a design discipline. It goes on to outline the previous development of critical systems development research in Scandinavia; based on this outline, it finally discusses the basic features of system development research.

In his “No silver bullet” article, Brooks (1986) analyses computer systems development and software, by identifying the accidental and the essential properties of the field. Accidental difficulties can be solved by means of new tools and methods, but without dramatic gain, whereas the essential difficulties cannot be subjected to technical or methodical solutions. According to Brooks, the complexity of software is an essential property that is different from the (accidental) complexity of other fields of engineering and science. The essential complexity of software has two important aspects: the accelerating development of information technology, and the sociocultural constitution of software. Although the impacts of the maelstrom-like development of information technology seem to be overestimated these years, it is important to keep in mind that the development of, e.g. the price/performance ratio of computer hardware radically exceeds what has been seen in other fields. This contributes to the complexity of software because basic software elements have no time to stabilise before the next innovations and new areas of application come into play; the bit tends to be the only stable unit in software. The accelerating development prohibits the development of computer systems development as a craft, because the needed software knowledge is constantly changing, and thus cannot crystallise into the kind of traditions that are the backbone of the development of the traditional crafts.

Software is basically a socioculturally constituted phenomenon, not a technically constituted one. Basic features of software are constituted in the context of use; in most cases, software has no meaning as isolated technical constructs (Andersen 1990, Bødker 1991). Thus, systems development research and praxis cannot be based solely on the systematic application of quantitative software measures or other methods derived from ideal natural science, but has to include a strong basis for understanding psychological, social and cultural phenomena. Furthermore, it has to comprehend development as a basic feature rather than as an exception. In addition, Brooks (1986) points to the methodical difficulties in finding comparable

cases for collecting statistically significant empirical data on systems development; systems development research has to rely on qualitative methods in the study of particular projects, organisations, events, etc. Such a discipline is by nature a pragmatic one directed to establishing relevant design knowledge, rather than establishing the universal, disinterested, but also irrelevant “truth”.

Thus, systems development research and praxis are not purely technical fields that can be based on algorithmics and quantitative measurements; systems development is rather, as suggested by Kapor (1991), a design discipline, which means that it is “concerned with making artifacts for human use” (ibid. p. 4). Due to the accelerating development of information technology, computer systems development cannot be expected to stabilise as a craft based on traditions, instead the field has to become an intertwined field of praxis, research, and theory, i.e. not separated into a design praxis and a basic science. Since the basic problems of computer systems development today are the accelerating development of basic technology and the sociocultural constitution of software, such an integrated field must be based on a theoretical framework that understands these features as basic conditions of the world rather than unusual exceptions. Such a theoretical framework has to be exclusive to avoid the multi-paradigmatic babel of conflicting world views, but at the same time it has to assimilate other perspectives and disciplines in an operational manner. Activity theory could be a possible, although not unproblematic, candidate for such an overarching framework (see also Bertelsen 1997b).

The approach to software development implicitly criticised by Kapor (1991) and Brooks (1986), perceives software as a pure technical matter to be optimised along objective criteria in order to achieve efficiency and to maximise revenue, thus completely neglecting issues like quality of working life. Bansler labels this approach “the system theoretical school” (Bansler 1987, 1989), Hirschheim and Klein (1989) refers to it as functionalism. This approach to systems development has been widely criticised for being both inefficient and inhuman, and alternatives have been formed including socio-technical as well as more critical approaches. In Scandinavia, critical approaches to system development research have been through three phases. The first phase, NJMF, DEMOS, DUE (see Ehn & Kyng, 1987, Bansler 1987) was the phase of politically engaged worker-academics cooperation; politically engaged computer scientists used their technical knowledge for the benefit of the workers and for general democratisation and improvement of the quality of working life. In these early projects the focus was on changed motives for engaging in the design of computer artefacts; from optimising productivity to serving the working class. In the next wave, it was realised that the new motives for engaging in system development created a need for new ways of doing design. Thus, e.g. the UTOPIA project (Bødker et al. 1987), developed new methods and techniques for specifying new com-

puter support for work and for involving users in the design process. In this second wave, the external political critique of the first wave was transformed into an internal and theoretical critique. This wave still dominates participatory design. In the third, wave it becomes clear that the new ways of doing design induce a need for new artefacts mediating the making of the actual computer systems. This can be seen in the Devise project (Grønbæk & Knudsen 1992, Kyng 1991)² where a focus on the development on tools and environments is taken. The new design life requires a new house. The third wave can be seen as the crystallisation of the new ways of making and doing design into deliberately formed new design artefacts. The APLEX (Bødker et al. 1988) expresses this new direction. In this line of development it becomes natural to use the artefacts mediating design as the vantage point for research on design as done in the design artefacts approach introduced in this thesis.

Systems development research is a field, based on a multiplicity of research methods and strategies, including intervention, theorising, field studies, and controlled experiments. Systems development research cannot, as it is possible with the natural sciences, be arranged according to the basic versus applied, and the theoretical versus experimental distinctions. It is impossible to make experiments detached from theoretical considerations which in turn only make sense when related to concrete reality. Research that is not contributing to the development of theory is not research, but mere consulting. However, consulting is in many cases a fruitful method, because the researcher has a chance to get close enough to the object of research (change processes) to see what is going on.

Systems development research is complicated by various features, setting a demand for the application of a broad spectrum of modes of enquiry.

- The domain of systems development research includes human beings, which are very hard to study in the controlled manner, applicable for purely technical phenomena. The objects of the ideal natural sciences are phenomena with causality, whereas phenomena involving human beings are intentional. In systems development research the researcher has to ask “why”, and results are always subject to interpretation. Quantitative methods are less applicable, instead research is often concrete, particular, and qualitative.

² The Devise project, or Devise Centre for Experimental Systems Development, is an ongoing cooperation between the Petri nets group, the programming languages and environments group, and the systems development group within the Department of Computer Science at the University of Aarhus. Further information can currently be found on <http://www.daimi.aau.dk/DEVISE>.

- Important qualities of computer artefacts are constituted in the context of use. This means that we never know what we are building before the implementation is done.
- Systems development research has change processes as its objects, which researchwise is a methodical difficulty.
- Intervention is both part of the object and a necessary method for learning about the object.
- In systems development research we are actively engaged in change processes and doing research at the same time. We need to engage ourselves in change to establish the object and to get close enough to it to have something real on which to base research.

Chapter 3

A scientific basis?

By introducing the notion of dialectical materialism in three steps, this chapter aims at elucidating the forms of knowledge established by systems development research. Firstly, by paraphrasing the development of the philosophy of science in the 20th century, represented by three central positions, the recurring dilemma between relativism and cosmological objectivism in western thinking is illustrated. Secondly, the role of basic assumptions in systems development research is discussed based on Burrell and Morgan's (1979) scheme for classification of organisational paradigms, eliciting the danger in confusing descriptive taxonomies with norms for research. Thirdly, a solution to the described problems is outlined through the introduction of Marx' theses on Feuerbach, which form basis for dialectical materialism.

Truth and method in the philosophy of science

The problem of finding true knowledge has been a main concern in western thinking from the old Greeks and onwards. In the 20th century, the truth problem has been the theme of the philosophy of science, attempting to establish a principle of demarcation between science and non-science, sense and nonsense. Logical positivism, founded in the 1920ies in the Vienna circle, approached the problem by analysing the logical structure of the statements made by science, rejecting the meaning of question about reality and relevance (Carnap 1935). Critical rationalism (Popper 1973) was concerned with the scientific process, particularly the process of justification of scientific findings, thus prescribing the ideal process of scientific inquiry. Common for the philosophy of science is that it has attributed true science to the application of specific methods; e.g. the inductive method in logical positivism, or the hypothetical deductive method in critical rationalism. Method has been seen as ensuring that no matter who does research and what their interests are, a result will be good and clean as long as "scientific method" has been applied. The history of science has been perceived as a rational and cumulative development towards greater knowledge.

In the sixties, philosophers like Kuhn (1962) and Feyerabend (1975) approached the problem of understanding science by actually looking at the history of scientific inquiry, and found that the ideals produced by philosophy of science did not have much to do with actual history. This led Kuhn to

the concepts of normal science, paradigm and scientific revolution, rejecting also the belief in a cumulative history of science. Feyerabend, being more normative, argues for methodical pluralism.

“There is no special method that guarantees success or makes it probable. Scientists do not solve problems because they possess a magic wand – methodology, or a theory of rationality – but because they have studied a problem for a long time, because they know the situation fairly well, because they are not too dumb [...] and because the excesses of one scientific school are almost always balanced by the excesses of some other school [...] Basically there is hardly any difference between the process that leads to the announcement of a new scientific law and the process preceding passage of a new law in society”. (Feyerabend 1975, p. 302).

The above statement may be too radical. Truth is not universal and accessible, method is not a universal guaranty (or infallible nanny), but we may say that honesty, and the search for universal truth as a self-educating norm, characterises good science in general.

Further, as claimed by Kuhn, different scientific schools and traditions have different, often incommensurable ways of establishing valid scientific arguments, or proofs. To be scientifically valid is to conform to the established norms within the tradition. Kuhn’s argument can be radicalised by claiming that scientific validity depends on scientific community, which is part of a broader societal praxis. Thus the production of scientific fact must be subsumed under the criticism of societal praxis.

Feyerabend is a rebellion within the field of analytical philosophy and traditional (Vienna and after) philosophy of science. In that context, it makes sense to point out that no universal principles exist and that “the mature citizen” must make up his³ own mind and make his own decisions. What Feyerabend ignores is that science, and any other human endeavour, is historically situated in a societal context. Feyerabend realises that the purpose of science is not to acquire divine insight, that science serves profane purposes, but he does not analyse how science itself is part of the economy. He destroys all cultural intolerance by stating: “At all times man approached his surroundings with wide open senses and fertile intelligence, at all times he made incredible discoveries, at all times we can learn from his ideas.” (Feyerabend 1975 p. 307). However, he does not ask what “senses” and “intelligence” are, or how these human faculties are constituted.

³ Cowboys are usually men.

On basic assumptions

Research is based on sets of basic assumptions not questioned on a day to day basis. The validity of the laboratory experiment, at least as it is traditionally perceived, can be seen as an example of that (Chapanis 1967).

The classification and analysis of different approaches to the study of a given subject matter can be based on an investigation into the basic assumptions of the different approaches. This is the strategy chosen by Burrell and Morgan (1979) in an attempt to classify different schools in organisational theories. This is done by locating them in a two-dimensional classification scheme, with assumptions about the nature of social science on one axis and the assumptions about the nature of society on the other (figure 1).

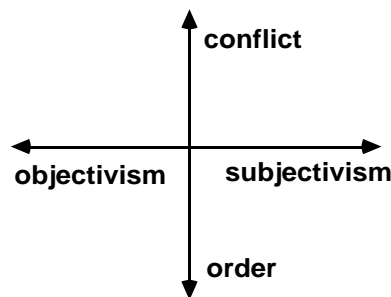


Figure 1: The Burrell and Morgan scheme.

The Burrell and Morgan scheme has a concrete historical scope and purpose as an instrument for making sense of sociological theories of organisation.

“In this analysis we polarise a number of issues and make much use of rough dichotomisations as a means of presenting our case. We do so not merely for the purpose of classification, but to forge a working tool. We advocate our scheme as a heuristic device rather than a rigid definition.” (Burrell & Morgan, 1979, p. xii).

The Burrell and Morgan scheme has also been applied outside the narrow context of organisational theory, e.g. by Hirschheim and Klein (1989) to the field of systems development research. This makes sense because systems development is also organisational change. A more problematic aspect of Hirschheim and Klein’s (op cit.) use of the Burrell and Morgan scheme is that they tend to use it as a normative rather than a descriptive tool. They understand the scheme as a systematic analysis starting with “a priori”, universal categories of scientific inquiry and the finite set of possible world views. Thus, they use the scheme as a finite and unchangeable space that any suggestion for a new theoretical framework needs to define itself in. By taking the dimensions to the extreme they end up degenerating the scheme into a “bad stuff - our stuff” dichotomy (figure 2).

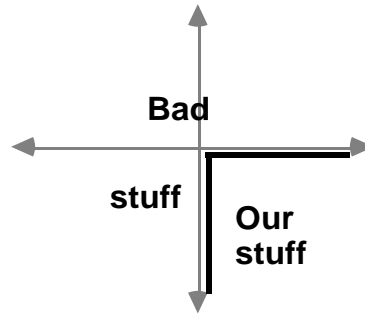


Figure 2: Hirschheim and Klein's degeneration of the Burrell and Morgan scheme.⁴

For the normative purpose a more viable strategy is to dissolve the dichotomies of the scheme and thereby to overcome the shortcomings of the paradigms laid out by the scheme. Then, the outcome of such an exercise may afterwards be analysed by placing it in the Burrell and Morgan scheme. As a normative instrument the order-conflict axis may still be valid; the litmus test is to ask whether hard emancipation is on the agenda. Dissolving the subjectivism-objectivism dichotomy, together with the interpretation vs. change dichotomy, is one of the central achievements of dialectical materialism, which is the basis for this thesis.

Dialectical materialism

As illustrated above, the Burrell and Morgan scheme may be a suitable tool for organising the analysis of concrete directions in organisational sociology, but as a general framework it suffers from the unfruitful ontological and epistemological dichotomisation between objectivism and subjectivism. The dialectical resolving of this dichotomy is one of the basic features of dialectical materialism, which is the basis for activity theory. Marx' notion of dialectical materialism, as expressed in the Feuerbach theses, is based on conceptions of history and dialectics developed by Hegel.

Hegel understands reality, including human reason and rationality, as being historically constituted. He perceives history as a spiritual development where human consciousness and knowledge are gradually enhanced in the attempt to appropriate reality and thereby consciousness by itself creates new cultural, technical and social institutions. By their consciousness, humans change the world, and spiritualise it according to their ideas. Through this transformation of the world things, the consciousness acts back on itself and is itself changed. The process and energy in the historical dynamics come from human consciousness. When consciousness appropriates the world through human work, the forms of consciousness are themselves ob-

⁴ This observation was originally made by Randy Trigg when Klein visited Århus in 1989.

jectified in the things in the world. This self-representation of consciousness in the world of things, is the basic figure in Hegel's philosophy of history. In this process, consciousness is alienated because it is split into something different from itself. Thus, for consciousness the purpose of history is to appropriate and spiritualise this different world, in which it has been alienated; when this world has been passed spirit can at last exist as itself. However, history does not in general coincide with the motives of the individual human beings. Thus, individuals become "tools" for the engrossing historical process. In this respect Hegel is in line with the enlightenment philosophical notion of the necessity of history. Humans are always part of, and contributing to, history, no matter what they do. Everything, even nature is subsumed under history. (Carlsen et al. 1984).

Hegel's philosophy is idealist, taking spirit as the first source of the world. In contrast to this, Marx emphasises the primacy of concrete material praxis.

"My dialectic method is not only different from the Hegelian, but is its directly opposite. Under the name of idea, Hegel turns the thought-process into an independent subject, which for him is the real demiurgos, whereas reality for him only is the outer, phenomenal form of the idea. With me, on the contrary, the ideal is nothing but the material which has been transformed and translated in the human head. [...] The mystification which dialectic suffers in Hegel's hands, does not prevent him from being the first to present its general form of working in a comprehensive and conscious manner. With him it is standing on its head. You must turn it right side up again, to discover the rational kernel within the mystical shell." (Marx 1962, p. 27, my translation).

Without entering into a philosophical treatment of dialectics and dialectical materialism, Marx' theses on Feuerbach (Marx 1845), are used here as an illustration of the basic ideas of dialectical materialism as basis for activity theory. On the surface, the theses deal with the sociology of religion, the question being: what are the origin and material status of religious ideas. They illustrate and condense important points where idealist philosophy, and cognitivism can be criticised. On a very compact form, they express the dialectical materialistic basis for activity theory (see below).

"1: The chief defect of all hitherto existing materialism (that of Feuerbach included) is that the thing, reality, sensuousness, is conceived only in the form of an *object or of something contemplated about*, but not as *sensuous, human activity, praxis*, not subjectively. Hence, in contradistinction to materialism, the active side was developed abstractly by idealism -- which, of course, does not know real, sensuous activity as such. Feuerbach wants sensuous objects—really distinct from the thought objects; but he does not conceive human activity itself as *objective* [gegenständlich] activity. Hence, in "Das Wesen des

Christenthums”, he regards the theoretical attitude as the only genuinely human attitude, while praxis is conceived and fixed only in its dirty-Judaic manifestation. Hence he does not grasp the significance of “revolutionary”, of "practical-critical", activity.” (Marx, 1845).

In this first thesis, Marx attacks the classical dichotomisation between, on one side a materialism (or objectivism) that deals exclusively with passive, or causal determination, in a mechanical way; and on the other side idealism (or subjectivism) that deals with human beings’ active, or intentional, relation to the world. The problem is that idealism is only abstract or spiritual whereas the former materialism is only mechanically concrete. Marx resolves this tension with a dialectical materialist concept of human activity, which is both objective and active. According to this concept human activity is practical-critical, or revolutionary in the sense that it changes or recreates its own objective conditions through practical actions.

The anti-Semitic sounding expression “dirty-Judaic manifestation of praxis”, makes sense if the anti-Semitism is filtered out. According to Wartofsky (1977, p. 319) it was commonplace in Germany at the time of Marx and Feuerbach, to identify egoism, practicalism, and utilitarianism with Jews. Thus, Feuerbach, according to Marx understands praxis as being short-sightedly this-sided, and he fails to acknowledge that the world can be changed only through the praxis of conscious human agency.

“**2:** The question whether objective [gegenständliche] truth can be attributed to human thinking is not a question of theory but is a *practical* question. In praxis, man must prove the truth of his thinking, i.e. the reality and power, the this-sidedness. The dispute over the reality or non-reality of thinking that is isolated from praxis is a purely scholastic question.” (Marx, 1845).

Mental activity (thinking) is material in the sense that it is impossible to talk about it as an entity isolated from practical activity. In particular, this thesis states that studies of cognitive processes isolated from concrete activity, as laboratory psychology (e.g. Card, Moran and Newell (1983) see below) tries to establish it, are absurd.

“**3:** The materialist doctrine concerning the changing of circumstances and education forgets that circumstances must be changed by men, and that the educator must himself be educated. This doctrine must, therefore, divide society into two parts, one of which is superior to society.

The synchronous changing of circumstances and of human activity or self-changing can be conceived and rationally understood only as *revolutionary praxis*.” (Marx, 1845).

Conditions for human activity can and should be changed by people themselves. While these conditions are changed, traditions and norms become obsolete; no universal standards exist above concrete practical activity.

Conditions and activity develop together; this coordinated process is the process of progress, emancipation, and revolutionary praxis.

The absence of universal standards above societal praxis, indicates that history and progress should not necessarily be understood as elevation through fixed levels of developmental advancement.

“8: Social life is essentially *practical*. All mysteries which lead theory to mysticism find their rational solution in human praxis and in the comprehension of this praxis.” (Marx, 1845).

By understanding human praxis, it is possible to understand social life in general. Ideas, theories, beliefs, etc. are parts of practical life and should be understood in that context. E.g. the capability of human beings to receive without giving back – receiving God’s grace like a child (Luke 18:17) – can be understood as prerequisite for the existence of an economy with considerable division of labour.

The human mind is societally constituted, rooted in the process of concrete societal history. Human beings are tied together by societal relations – by history not by the fact that they belong to the same species. Thus, human beings cannot be understood as universal abstractions, but must always be analysed in a concrete historical, and societal context. In the Feuerbach theses, Marx resolves the dichotomy between materialism that understands mental phenomena as mechanically determined by the material conditions, and idealism rejecting that the material world can have any influence on thinking; he shows that humanity is societally constituted as “the ensemble of the societal relations” situated in the course of concrete history; he resolves the separation between the subjective and the objective; and he shows that thinking cannot be separated from praxis.

In contrast to Hegel who advocated an idealist armchair philosophy in the shape of Minerva’s owl, Marx advocates a practical and concrete philosophy.

“11: The philosophers have only *interpreted* the world, in various ways; the point is to *change* it.” (Marx, 1845).

Chapter 4

Human Activity

This chapter outlines the theory of human activity, Star's notion of boundary objects and heterogeneity, and Wartofsky's concepts of primary, secondary and tertiary artefacts related to perception and action. These contributions are the basis for the understanding of design and design artefacts introduced later in this thesis.

From a simpleminded reductionist point of view, we can regard computer systems development as involving an assembly of human beings and technical artefacts being changed or re-constructed to satisfy some motivation. We may thus want to understand the individual components involved in order to understand the whole process; we need to understand computers and we need to understand the individual human beings involved. Understanding human beings as a basic unit is what psychology is about, thus we may want to start with a psychological perspective.

At the outset, activity theory is a dialectical materialist psychology aimed at understanding the mental capacities of a single human being. The important thing about activity theory is, however, that it rejects the isolated human being as an adequate unit of analysis, thus insisting on culturally and technically mediatedness. We then end at a unit of analysis which includes technical artefacts and cultural organisation. The main references for this chapter are Engeström's *Learning By Expanding* (1987), Engelsted (1989), and Mammen (1989).

According to Vygotsky (1978), Human activity has three fundamental characteristics; firstly, it is directed towards a material or ideal *object*; secondly, it is *mediated* by artefacts; and thirdly, it is socially constituted within a *culture*. Historically, activity theory is an answer to the problem of studying isolated individuals in the laboratory setting, e.g. pure memory. Instead of dealing with the isolated relation between the subject (S) and an object (O), from which the subject is perfectly separated, Vygotsky introduced a mediating X, which is culturally constituted. This mediating X is also referred to as instruments, which can be either technical instruments (tools) or psychological instruments (signs). Psychological instruments like language and concepts are internalised during child development, which is the reason why it is not possible to experiment with (or even to talk about) the basic, uni-

versal, unmediated, cognitive apparatus. Vygotsky distinguishes between meaning and sense in language⁵. Meaning is stable, and is what the sign points at or denotes, whereas sense is the fluctuating contents of the sign determined by the use of the sign in praxis. This doubleness does not cause many problems in the use of language.

Leontjev applied a slightly different approach in analysing cognition. In Vygotsky's group, Leontjev was assigned to the task of describing the development of natural history, from one-celled organisms to human beings. In describing this development the hunt becomes an important laboratory for thought. In natural history, the first important quantum leap in the development from cell to man is when animals start to work together in fulfilling their needs. Thus for Leontjev the basic triangle is not S-X-O, but the pre-human S-C-O where C is community.

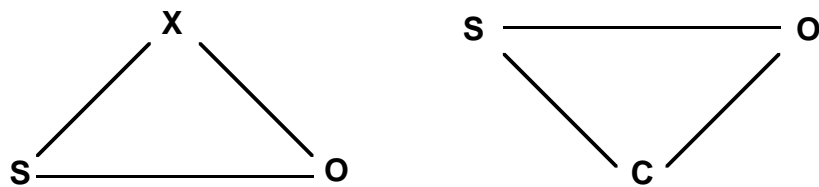


Figure 3: Triangles of activity, on the left human activity mediated by artefacts (Vygotsky), on the right socially mediated activity (Leontjev).

At the level of animals it is possible to identify embryonic forms of mediation of S-O relations in form of *ad hoc* tools, mediation of the S-C relation as emerging rules and rituals, and mediation of the C-O as emerging division of labour.

The quantum leap from animal to man takes place when these embryonic mediations become a solid part of the triangle of activity. According to Leontjev (1978), human activity can be analysed into a three-level hierarchy of activity, action, and operation, each of which reflect the objective world. Activity is directed to satisfy a need through a material or ideal object. The subject's reflection of (including expectation to) this object is the motive of the activity. Human activity is carried out through actions, realising objective results. These actions are governed by the conscious goals of the subject. Goals reflect the objective results of action. Actions are realised through series of operations; each "triggered" by the conditions and structure of the action. They are performed without conscious thinking but are oriented in the world by a non-conscious orienting basis. Goals that are different from the

⁵ I am indebted to Holger Hybshmann Hansen's review of the development of the concepts meaning and sense in Vygotsky's work at "8. Danske seminar om Virksomhedsteori: Virksomhed - betydning - mening" [The eighth Danish seminar on Activity Theory: Activity - Meaning - Sense].

motive, but still realising it are only possible in human activity; in animals goal and motive are always the same.

Activity	Motive (need)	Why?
Action	Goal	What?
Operation	Condition (structure of activity)	How?

Figure 4: The table adopted from Bærentsen (1989) shows the relation between the three levels of activity.

The three levels of activity are not fixed (figure 5); an action can become an operation through automatisation/internalisation, and an operation can become an action through conceptualisation in breakdown situations (Bødker 1991). A separately motivated activity in one context can be an operation in another activity. The focus in activity theory on how human acts transfer between the different levels of activity is a very important feature that distinguishes this framework from the mainstream of cognitive theories, e.g. Card et al.'s (1983) engineering psychology, where acts are classified as belonging to static categories, e.g. time bands (Bannon & Kuutti 1993). In short, development is a basic feature in the framework of activity theory.

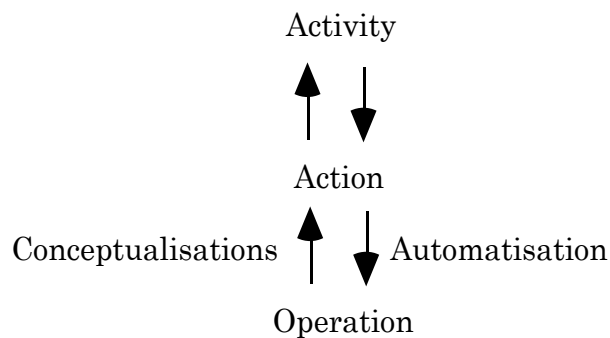


Figure 5: The dynamic relation between levels of human activity.

Leontjev's notion of human activity can be depicted as embedded triangles (figure 6), which is the Subject-Object-Community triangle of pre-human activity expanded with societally constituted forms of mediation: instruments, rules and division of labour (Engeström 1987). The specific form of the triangular figure is not very important. The important thing is that activity is an intertwined system; if one corner changes the system becomes unstable and must develop to obtain renewed stability.

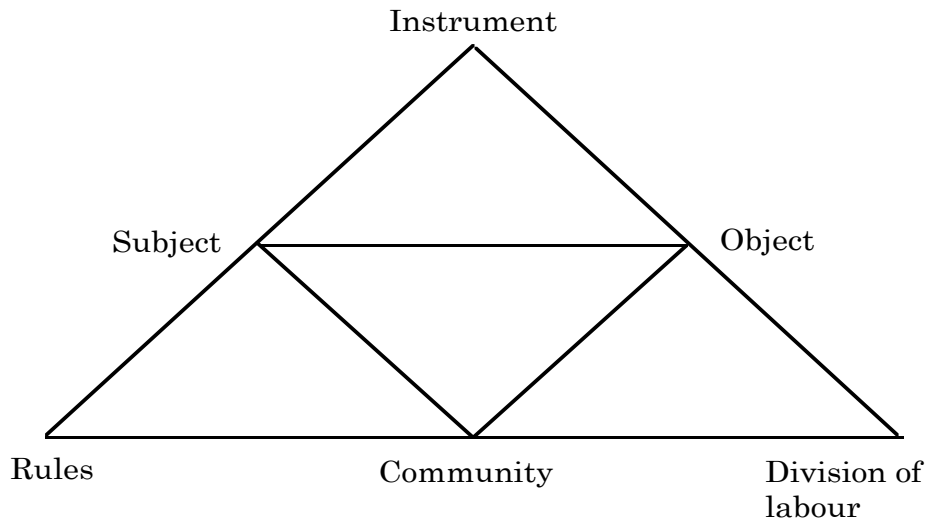


Figure 6: Leontjev's theory of human activity as depicted by Engeström (1987)⁶.

Activity is constantly developing as a result of instability (see below) and due to the construction of new needs. Activity is historically crystallised into artefacts; in this sense the historical development of activity can be read from the development of artefacts mediating the praxis (Bærentsen 1989, Bannon and Bødker 1991).

“Artifacts can be characterised as *crystallized knowledge*, which means that operations that are developed in the use of one generation of technology are later incorporated into the artefact itself in the next” (Bannon & Bødker 1991, p. 243).

Activity is crystallised into artefacts in two ways. Firstly, as externalisation of operations with earlier artefacts, and secondly, as secondary artefacts representing modes of acting in the given activity (see below section on perception and action.). Artefacts mediating human activity are not just more or less suitable attachments to human praxis, but they are constituting activity. Thus mediating artefacts can be regarded more stable than the object of activity. This points to the argument (given below) that modelling in design (e.g. object-oriented modelling) should be understood as the transformation of artefacts mediating the domain of use. Crystallisation can be illustrated with an example from Mathiassen (1981), who approaches the problem with a different terminology. The basic concepts are work process, technique, and tool.

“Take for instance the relation between programming techniques and programming languages: In the beginning when programming (the work

⁶ In the age of 3D graphics it could be tempting to draw the figure more like a tetrahedron.

process) was done directly in absolute machine language (the tool), it was necessary to develop a conscious relation (techniques) to the use of individual memory cells; otherwise it would come to grief. Later symbolic machine language was developed (new tool), which made it possible to use the computer as a help for administering the use of physical memory, and which at the same time formed a basis for new programming experience. The development of techniques and tools goes hand in hand". (Mathiassen 1981 p. 100, my translation).

In terms of activity theory, administering memory usage in programming pure machine language was crystallised into the next generation of programming artefacts, symbolic machine language.

Engeström (1987) develops the analytical framework of activity theory to include neighbouring activity systems. The basic idea is that the components of a given activity system are produced in other activity systems, and that the object of activity is consumed in other activity systems. Instruments are produced in the instrument producing activity, subjects are shaped in the subject producing activity through education, etc. The relation to the neighbouring activity systems in the model is introduced through the classification of contradictions in activity. In general, contradiction is the source of development, empirically contradictions appear as disturbance.

According to Engeström (1987) there are four types of contradictions: primary, secondary, tertiary, and quaternary.

The **primary contradiction** is the contradiction of commodity between use and exchange value. This double nature is a basic feature of the economic structure in capitalist culture, and it penetrates every single corner of the triangle and is the basic source of instability and development (see also Marx 1962, pp. 49-55).

The **secondary contradictions** of an activity system are contradictions between the corners of the triangle, e.g. between the skills of the subject and the instrument, or between rigid rules and new flexible instruments.

Tertiary contradictions are contradictions between the central activity and new motives of forms of activity. In "Learning by Expanding", this contradiction is generated by a culturally more advanced activity, by representatives from this other culture introducing culturally more advanced objects or motives into the central activity.

The obvious problem is to know what is more advanced without subscribing to a very deterministic idea of history. However, the concept of a culturally

more advanced activity does not necessarily imply historical determinism⁷; it can also be interpreted as potential ways of conducting the central activity differently. Thus, the conflict between the historical determinism, still present somewhere in the basis of activity theory, and notions of heterogeneity (e.g. Star 1989) is not necessarily insoluble (see below).

Quaternary contradictions are contradictions between the central activity and the neighbouring activities; i.e.: 1) object activities where the immediate objects of the central activity are embedded, 2) instrument producing activities where key instruments of the central activity are produced, 3) subject producing activities educating the subject of the central activity, and finally 4) rule producing activities like administration and legislation. An example of a quaternary contradiction is the contradiction between education of computer scientists at universities, focusing on mathematical formalisation, and the central activity of computer scientists working as system developers in the industry. The quaternary contradictions show that activity systems cannot be understood in isolation.

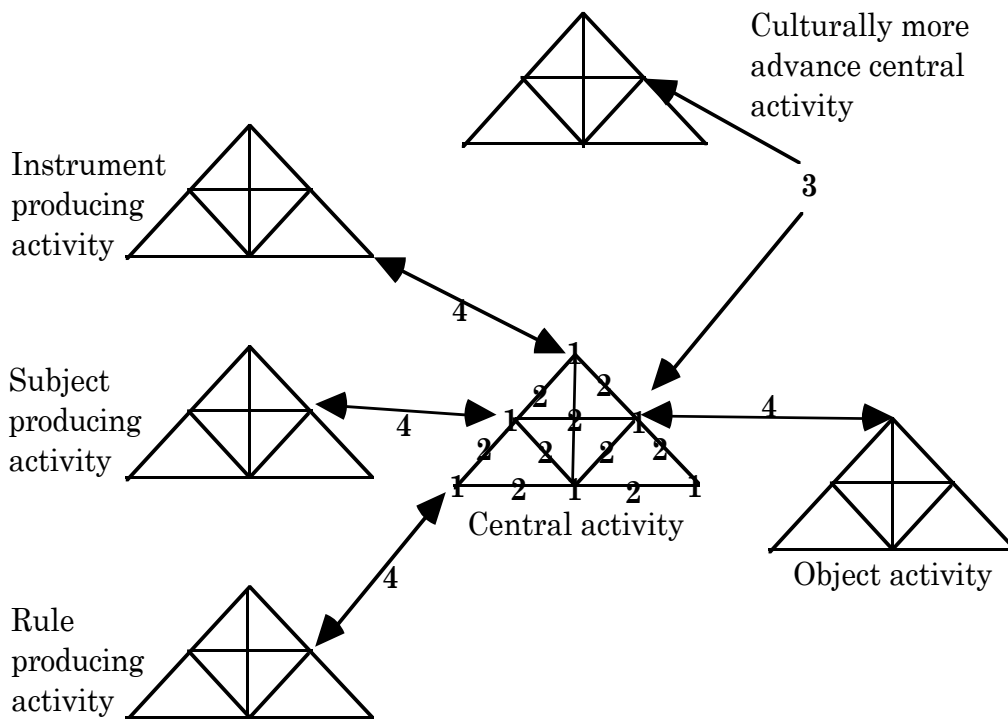


Figure 7: The relation between activity systems in terms of classes of contradictions.

⁷ The idea that history will proceed through an unavoidable sequence of phases and end in a determined stage, communism.

In applying an activity theory based analysis to a systems development project, it is not always obvious how to select the central activity. E.g. in the Music Festival project, it was not at all clear whether the work of the half-time employed sound and light person, the work of the entire sound and light group or maybe somebody else's work, should be considered the central activity. In most situations, several possible central activities will be visible.

Naturally the 'neighbour activities' also include central activities which are in some other way, for a longer or shorter period, connected or related to the given central activity, potentially hybridizing each other through their exchanges (Engeström 1987, p. 88).

Rather than dealing with one fixed central activity, we are often confronted with a number of intertwined activities each in different ways are potential central activities. In the end identifying the central activity is a matter of choice, a matter of how we are going to act in changing the world, a matter of the purpose of the analysis. Analysing praxis in terms of interrelated activity systems provides a dynamic analysis, making it possible to understand specific acts differently according to viewpoint, but maintain a materialist view on the situation. According to Bødker & Mogensen (1993), this is an important advantage of activity theory over static notions of mechanisms based on the concept of articulation work.

A technique in identifying central activities in chaotic empirical data, is to work "upside down", by botanising for contradictions and then classify them into primary, secondary, tertiary, and quaternary, and subsequently identify of one or more interesting central activities. This was done in the analysis of the festival project (Bertelsen 1996a).

Contradictions as a perspective in systems development research

The concept of contradiction or tension alone can be a fruitful vehicle for understanding system development and praxis. In (Bertelsen 1996a), Engeström's classification of contradiction is used to analyse the Music Festival case; this gave rise to a series of interesting observations which had not been considered earlier in the project. In general, the existence of antagonistic contradiction has been a key premise in the Scandinavian collective resource approach, which has made strong practical focus on the conflict between capital and wage labour, and how this conflict can become a resource (Borum & Enderud, 1981).

The Norwegian left-wing seems to have a persistent Maoist orientation, which is reflected in Norwegian critical systems development research by contributions (Bjerkness 1992, Bratteteig & Øgrim 1994) based on Mao's notion of contradiction (Mao, 1937). In this notion everything tends to be contradictions, but there will in any situation be a principal contradiction, e.g. the contradiction between social classes. In a contradiction there will be

one side dominating the situation. This Maoist concept of dialectics is basically a vulgarised version of Hegel's dialectics. The weakness of the Maoist approach is that:

“Mao does not give any frame for discussing and understanding the contradictions. In this way the notion may become very subjective”.
(Bjerkness, 1992).

To put actual flesh and blood into the analysis, the Norwegian approaches have combined the Maoist notion of dialectics with elements of Soft Systems Methodology (Checkland 1981) in an approach referred to as “Soft Dialectics” (Bratteteig & Øgrim 1994). The problem in doing so is that the basic assumptions of SSM may take over the role as a main perspective, because the Maoist concepts are not transferred into an operational orienting basis for the praxis of “Soft Dialectics”. In the treatment of theories as design artefacts below, it is argued that in the absence of an explicit world view, the world views embedded in the applied design artefacts will take over the situation. The example of “soft dialectics” shows that the world view not only should be explicit but also has to be coherent and to some degree operational. Thus, the advantage of basing design and research on activity theory is that this approach builds on a complete and coherent set of assumptions about the world, in particular that activity theory, in contrast to SSM, understands the creation of new motives as a basic feature. Activity theory, however, does not make the use of techniques like rich pictures from SSM impossible.

human-computer interaction and computer mediated activity

Human activity mediated by computers (i.e. the object of our design activity) has been dealt with within the field of human-computer interaction. The basic understanding put forward in here shapes the process of design and the possible design solutions (see discussion on theories as design artefacts below and in (Bertelsen 1994b)).

The traditional perspective on human-computer interaction in the early eighties (e.g. Card et al. 1983) was based on the mechanical reductionism of cognitive science, i.e. human-computer interaction was understood as a symmetrical relation between two machines and the aim of the discipline was to tune the computer side, or more specifically the human interface, to reach an optimal fit between the two entities.

The lack of practical success for the orthodox cognitivist approaches gave rise to contributions taking the user's daily work more directly into account (Norman and Draper 1986, Carroll et al. 1991). However, these approaches did not reject the mechanist assumptions systematically; this created a sort of mismatch between the intentions of research and design recommendations, and the basic theory underlying the endeavour.

With the introduction of activity theory as a basis for human-computer interaction research and praxis (Bødker 1991), the field has acquired a theoretical grounding matching the relevant practical research problems. With activity theory it is possible to maintain both the original focus on interaction between human and computer, and notions of context etc. — the interface explodes while it continues to be the pivotal point of research.

The otherwise vague concept of “user interface transparency” can become a well-defined concept in terms of activity theory by observing that the interface (or rather the entire computer artefact) is transparent when, and only when, the user is handling the computer artefact at the level of operations (Bardram & Bertelsen 1995). In computer mediated activity, the situation is slightly more complicated than in the case of a carpenter using a hammer to drive a nail; in that case “transparent hammer-carpenter interaction” occurs when the carpenter is able to focus on the nail. The complication in the case of computer mediated activity is that the object of activity in most cases is inside the computer: E.g. the text which is the object of writing with a word-processor is represented as bits in the same RAM chip that holds the word-processor program. Bødker (1991) analyses this problem in detail by distinguishing between physical aspects, handling aspects, and subject/object-directed aspects of the interface. In the situation of total breakdown (Winograd & Flores 1986) the user directs actions to the physical aspects of the computer, by e.g. trying to pull out a diskette stuck in the drive or searching for the power switch to turn the not working machine on. The handling aspects are parts of the computer artefact through which the user interacts in the normal situation of use; these become the object of action, e.g. when the user learns to use a new program. The subject/object-directed aspects in the computer artefact, can be representations of subjects or objects outside the computer, or objects that exist only inside the computer. These aspects are equivalent to the nail in the hammering case.

Development

Vygotsky’s theory of human activity is a psychology with a strong focus on developmental psychology and pedagogy. The concept of the zone of proximal development is aimed at changing the focus of developmental psychology (which is in practice a foundation for teaching strategies) from already acquired skills to potential skills “waiting” to mature in the individual. The concept is defined by Vygotsky as:

“the distance between the actual developmental level as determined by independent problem solving and the level of potential development as determined through problem solving under adult guidance or in collaboration with more capable peers” (Vygotsky 1978 p. 86).

Learning, in this framework, is a voyage through the zone of proximal development. The concept of the zone of proximal development has been widely

applied outside the areas of pedagogy and developmental psychology, and has a central role in Engeström's framework of expansive learning. In general the concept of the zone of proximal development is interesting because it indicates that theory of human activity treats potentiality and development as basic aspects of human activity. People are not understood in terms of what they are but in terms of what they are becoming. Furthermore, it is important to notice that the zone of proximal development is a concept of social mediation of activity and of development of mental capacities.

In (Bardram & Bertelsen 1995), we used the concept of the zone of proximal development in the context of development of transparency in the one user one machine situation, which at first sight appears to be in contradiction with the concept since there does not seem to be any social mediation of the development of this situation. However, computer artefacts are not only tools mediating users' relations to their object of work, they are, at the same time, media mediating the relation between designers or culture and users. Computer artefacts are social mediation in the same sense as books. By careful planning the designer can place clues in the interface affecting users at specific stages of the development, and in this way build a curriculum into the artefact.

In the original form, the concept of the zone of proximal development is dealing with skill acquisition, thus it contains a vertical notion of development in that the learner is moving from one state of activity to a more advanced state. Engeström partly criticises this aspect of Vygotsky's concept of the zone of proximal development by stating that:

“Vygotsky's concept of the zone of proximal development is itself in need of development. The cultural-historical school founded by Vygotsky has up to the present time concentrated on the acquisition, assimilation and internalization of the tools and sign systems of the culture. How these tools and sign systems are created has mainly been treated as a problem for the future” (Engeström 1987 p. 172).

Engeström concludes that the chief defect of all hitherto existing concepts of zones of proximal development is that they do not take into consideration the development of the total activity systems and their motivations.

“What is not discussed is whether and how *the activities themselves as social systemic formations develop and change constantly*” (Engeström 1987 p. 173).

This is setting the program for his own development of activity theory and results in a reformulation of the zone of proximal development.

“A provisional reformulation of the zone of proximal development is now possible. It is the distance between the present everyday actions of the individuals and the historically new form of the societal activity

that can be collectively generated as a solution to the double bind potentially embedded in the everyday actions” (Engeström 1987, p. 174).

The voyage through the zone of proximal development can be described as an expansive cycle of five stages (figure 8). According to Engeström (1987) the starting point of the voyage is a state of need in the central activity, which in general originates from the primary contradiction between use and exchange value. In the second phase—the double bind phase—the central activity is faced with insoluble dilemmas caused by secondary contradictions between corners of the activity caused by the introduction of new corner elements. In the third phase, motive/object construction begins with finding instruments that can function as “springboards” for initially breaking the double bind and for initiating the constructing of a general model of the reformed activity. New instruments and actions for the reformed activity are constructed through modelling. In the fourth phase—the application and generalisation phase—the new instruments are confronted with the old activity; precursors of the new activity, the expected new and the old activity are conflicting and disturbing each other, and the societally new is born in forms not anticipated in the earlier phases. In the last phase, the new activity is consolidated which may happen through three sub-phases. Firstly, the instruments are systematically applied in a repetitive and explicit way; secondly, the use of the new instruments is varied, and the new activity is integrated into the entire activity system. In the third phase the new activity takes effect in the relation to the neighbouring activities. The last phase is characterised by quaternary contradictions, “the new central activity has to compete with and adjust to the dynamics of the neighbour activities” (Engeström 1987, p. 191). Thus, activity 2 is not a stable finite state and development will continue through recurring cycles of expansion.

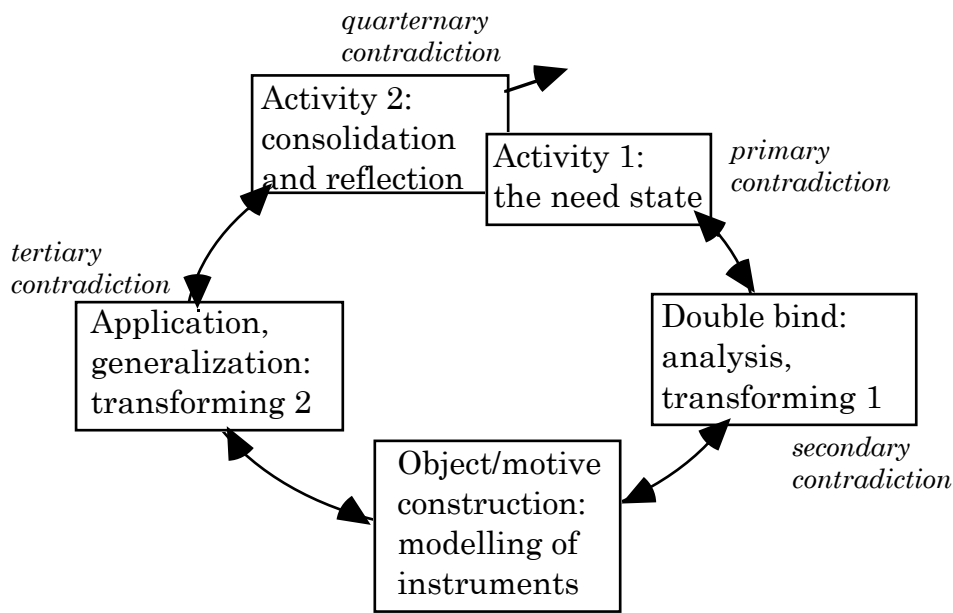


Figure 8: The phase structure of the zone of proximal development depicted as the general cycle of expansion (adopted from Engeström 1987, figure 3.3).

Based on the cycle of expansive transition Engeström (1987) outlines a methodical cycle of expansive developmental research. The steps in this cycle are: 1) Phenomenology and delineation of the activity system. 2) Analysis of activity: object historical, theory historical and actual empirical. 3) Formation of new instruments: finding a springboard, formulating general instrumental model and its derivative models, formulating a microcosm for turning the instrumental models into new forms of praxis. 4) Practical application of new instruments to change the activity through strategic tasks. 5) Reporting and evaluation.

The isomorphism between the cycle of expansive developmental research, the cycle of expansive learning, and Vygotsky's methodological scheme points to basic similarities between development in children, development in collectives, and developmental research, but it also hides important differences; e.g. between reporting after the process has ended and the uncertainty of the consolidation phase. Developmental research is more related to praxis than to, e.g. the hypothetical deductive method (Popper 1973) of idealised natural science.⁸

⁸ Engeström's methodological cycle of developmental research is isomorphic with the expansive cycle, in the same way as Fitts' law (Fitts' 1954, see also Bertelsen 1994b) states that human motor performance is isomorphic with Shannon's (1949) mathematical model of communication. Engeström's cycles are based on folk tales and stories about subjects in a need state going on a voyage to find a solution and then returning to (a renewed) home, and the resulting model is so simple that the pure morphology (partially derived from Scribner's illustration of Vygotsky's four steps) alone doesn't reveal anything.

Engeström's (1987) reformulation of the zone of proximal development and the methodology for expansive learning transcends the limiting focus on individual assimilation into established activity forms of Vygotsky's original concept. But it does still conform to the uni-directionality of vertical development, as it is expressed in the concept of "the culturally more advanced activity". This reflects the inherent tendency of dialectical materialism to subscribe to historical determinism, and a general uni-directionality of western thinking in the enlightenment tradition. Such an uni-directionality is a problematic foundation for systems development and for systems development research, because it deforms the mutuality of the development process; it will cause us to understand the computer professionals as "the more capable peer" in the heteropraxiality (see below) of design, setting narrow limits for the outcome of the development effort. In a later paper, Engeström (1996) formulates three challenges to developmental theories (Vygotsky and Piaget), which programmatically outlines the next (ongoing) leap forward of activity theory, away from the uni-directionality of historical determinism. These challenges are:

"(1) instead of just benign achievement of mastery, development may be viewed as partially destructive rejection of the old; (2) instead of just individual transformation, development may be viewed as collective transformation; (3) instead of just vertical movement across levels, development may be viewed as horizontal movements across borders." (Engeström 1996, p. 126).

The second, and to some extent the first, challenge are reflected in (Engeström 1987); thus it is the formulation of a horizontal concept of development that forms an important agenda for further development of activity theory. The three leading concepts in formulating a new developmental theory are contradiction, zone, and mediation. Engeström points to contradictions as a fundamental feature of "the world", not merely a result of cognitive mismatch (as suggested by Piaget). Zone is different from Vygotsky's conception of the zone of proximal development in that it is not necessarily a room for elevation across levels through meeting the more advanced, but rather a room for meeting the different, the incomprehensible mediating horizontal development. The role of mediation is that the mediating artefacts through re-mediation lead to new goals and actions.

"The process described by Høeg [Engeström uses a novel by Peter Høeg as a vehicle for the argument /ob.] calls for a conceptualization of mediation as more than technical amplification. It calls for studies of arti-

A challenge for further research would therefore be to analyse the developmental stories used by Engeström with structuralist oriented models of folk tales and literature, e.g. Propp or Greimas; and then in turn use the outcome of such an analysis as a basis for a methodological model of developmental/action research.

facts mediating in the construction of new tasks, in the formation of motives, and in related developmental shifts” (Engeström 1996, p. 131).

Engeström (1996) concludes that a developmental theory that is able to explain something about significant transformations in human life, will not be a combination of the two main perspectives outlined, but the two perspectives maintained in isolation but also transcended.

Activity theory yields a framework for understanding human praxis as historically constituted and mediated by artefacts, setting the agenda precisely at central issues in systems development research. With the ongoing development of the framework, activity theory is about to get rid of its detrimental historical determinism, although it is unclear if the new orientation toward heterogeneity, as expressed in the concept of horizontal development, will turn the framework into inconsistent theoretical babel. Only the future will reveal how deep the revision needs to be. Also induced by the ongoing development of AT is the relativism problem caused by the absence of a unidirectional notion of development and progress. We have to ask whether any change, caused by meeting what is different, is a desirable development; and how we distinguish. To avoid falling into relativism, activity theory needs to develop a new concept of progress.

Activity theory and systems development research

Currently, activity theory is catching a lot of attention in the fields dealing with the design and use of information technology. Despite this fact, very little work substantially based on activity theory have been presented. Based on experience from the UTOPIA project (Bødker et al. 1987) Bødker's thesis from 1987 (published in Bødker 1991) focuses on how quality of use develops in use. Christiansen (1988) focus on the development of knowledge in systems development, and on the role of different conceptions of people and technology in relation to the development process. Based on Engeström (1987), Bisgaard et al. (1989), present a framework for the utilisation of conflicts as a resource in systems development; they furthermore apply this framework in the *post hoc* analysis of two systems development case stories. Kuutti (e.g. 1991) has bases his work on a very thorough application of Engeström's framework in the fields of information systems, computer supported cooperative work, and human-computer interaction (Bannon & Kuutti 1993). Not directly related to systems development research, Raeithel (1991) outlines activity theory as an alternative to Winograd & Flores' (1986) critique of artificial intelligence, which according to Raeithel reduces man to “a speaking animal that loves to bargain” (p. 392). The book “Context and Consciousness” (Nardi (ed.) 1995) was intended to remedy the absence of practically applicable methods based on activity theory; unfortunately the book contains only a few operational contributions.

Heterogeneity and artefacts

Activity theory takes activity systems as the basic unit of analysis, primarily understanding the situation in terms of uniform motives. When several activity systems are involved, they are understood as having the same kind of rationality and the same direction of development, although some more advanced. This does conflict with the lessons learned about the necessity of doing prototyping with users. In such situations, both parts (users and designers) will learn, but they are radically different to a degree that they will never understand each other. Thus, we are in need of a concept that can unify the design situation without subsuming everything under the same rationality and the same line of development.

Recent development of activity theory has been influenced by north American notions of heterogeneity, e.g. from symbolic interactionism. The concept “boundary object” (Star 1989) is particularly useful in understanding the design of computer artefacts and in particular the artefacts mediating design. The concept boundary objects is based on the study of scientific work, and is aimed at understanding how the multiple and highly diverse viewpoints of actors participating in the scientific process are coordinated. The cases analysed by Star (1989) are a group of neurologists in England at the end of the last century, and the people working on a zoological museum in California during the first three decades of this century⁹.

Systems development is similar to the scientific processes described by Leigh Star in that they involved a heterogeneous group of people who work together in relation to a material object that matters in varying degree for the participants, and which is perceived in different ways.

According to Star (1989), boundary objects are objects used by different parties in different localities. They are at the same time robust enough to maintain their identity across use by different parties, but also plastic so that they can adapt to the constraints and needs of the different parties working with them. Across sites, boundary objects are weakly structured, but in a specific context their use become highly structured. Thus, boundary objects mediate the relation between actors with divergent viewpoints. The concept of boundary objects focuses on information and representation of information. Star explains that it is important that there are different kinds of boundary objects, depending on the characteristics of the heterogeneous information joined. “The combination of different time horizons produces one kind of boundary objects; joining concrete and abstract representations of the same data produces another” (Star, 1989, p. 47).

⁹ Obviously these studies have been based on archivals.

Based on the studies of neurologists and the zoological museum, Star identifies four types of boundary objects. “Repositories” are ordered piles of objects indexed in a standardised and modular way, e.g. a library or a museum. Repositories are dealing with heterogeneity caused by differences in the unit of analysis, e.g. when the amateur zoologists collect individually interesting beetles whereas the biologists are interested in what the same beetles represent. “Ideal types or platonic objects” do not describe the detail of the concrete, but are adaptable abstractions like maps and atlases. “Platonic objects arise with differences in degree of abstraction such as those that obtain in the clinical/basic distinction. They result in the deletion of local contingencies from the common object, and have the advantage of adaptability.” (Star 1989, p. 49). “Terrain with coincident boundaries” are common objects with shared boundaries but different internal contents. Star’s example is the outline of the state of California which is filled out in different ways by amateur zoologists and professional biologists working together. Terrain with coincident boundaries allow different groups to autonomously work with different means of aggregating data. “Forms and labels” are aimed at mediating common communication across distributed work groups. They ensure that information is collected in a uniform and standardised manner no matter who does it, local uncertainties are deleted. Forms and labels are, according to Star (1989), what Latour calls “immutable mobiles”, i.e. objects that maintain unchanged information when transported over distance. These four types of boundary objects are highly dependent on the specific empirical material Star refers to. A more generalised framework would merge some of these types and add others. However, in the context of this thesis these types serve as examples of boundary objects, and they illustrate that different kinds of heterogeneity are handled through the use of different kinds of boundary objects.

Boundary objects are relevant in a design context because the artefact mediating the design process in many cases are used by different groups with different roles, motivations, and professional backgrounds. A context diagram (Yourdon, 1982) can be understood as a “platonic object”; the domain of use can be seen as a “terrain with coincident boundaries”; a library of reusable software components is a “repository”; some design methods, e.g. SPU (Biering-Sørensen, et al., 1988) focusing on the production of specific documents during the design process, present themselves as “forms and labels”, but are not so in concrete praxis (see interview with designer of embedded software below).

Boundary objects can be compared to the Wittgensteinian notion of language games used heavily by, e.g. Ehn (1988). Language games are used by Ehn to explain the process of creating mutual understanding through co-creation of language. Meaning is not universal but attributed concrete praxis. What is accomplished during the language game of a design workshop is manifested through the creation of design artefacts which makes sense to

the heterogeneous group of participants because they are boundary objects. Language itself is a boundary object. The two concepts complement each other. Language game is a process oriented concept, whereas boundary object is emphasising the artefacts structuring the process.

Perception and action —classes of artefacts

As described in the section above, the basis of activity theory is that human beings' relations to their surroundings are mediated by artefacts. Vygotsky's concept of psychological instruments emphasises that this mediating is a basic feature – the bare human being without instruments does not exist.

Action¹⁰ and perception are mediated by artefacts in a dialectical way; action is perception and perception is action. A hammer mediates changing the surroundings by driving a nail in a board, but at the same time it mediates the carpenter's perception of that particular piece of wood. All artefacts possess this double character of mediating both perception and action, although in many situations it is possible to identify a main direction of the relation. Driving a nail with a hammer is mostly action; examining a blood test with a microscope mostly perception. The separation of perception from action is prominent in many disciplines related to the design of computer support for work. It has been the core of the traditional work sciences as founded by Gilbreth (1911) and Taylor (1916) — the stopwatch man evaluates operations and the worker performs operations. Card et al. (1983), along with the mainstream of mechanist cognitive science, hardwire the separation into the cognitive apparatus by “constructing” separated perceptual, cognitive and motor processors. Newer revisionist accounts like Norman's (1991) concept of cognitive artifacts maintain the separation, thus failing to explain how tools mediate our understanding of the world.

In constructing a notion of design artefacts, a more differentiated understanding of artefacts is needed, especially the representational aspects of artefacts. Marx Wartofsky (1973) introduces a distinction between primary, secondary and tertiary artefacts which seems to meet this need. According to Wartofsky (ibid.), classical theories of perception suffer from three important defects. Firstly, rationalist as well as empiricist theories understand perception as an ahistorical, universal characteristics of the species. Secondly, most accounts are based on seventeenth-century psychological models of perception which only deal with anomalous situations related to

¹⁰ The word action in this section is not used in the strict activity theory meaning, but as outward human behaviour.

geometrical science; they do not see the historical limitations of their own theories. Thirdly, they understand perception only as an inner process. Wartofsky, in contrast to this, advocates a historical epistemology, understanding perception as an integral part of praxis (not a prelude to action); perception changes historically together with that praxis, and it is determined by and help determine changes of praxis¹¹. Thus, the theory of perception suggested by Wartofsky is characterised by having the following three features: 1) Perception is understood as being historically variable; 2) Rejection of the seventeenth-century psychological model of perception; and 3) Perception is taken to be a mode of outward action. In agreement with the theory of human activity described above, Wartofsky understands the perceptual “apparatus” as involving historically developed artefacts. The distinctive human form of acting is constituted by the creation and use of artefacts, in reproducing the species as well as in producing the means of existence. Immediately Wartofsky identifies two types of artefacts, primary and secondary. *Primary artefacts* are used directly in production. *Secondary artefacts* are used in preserving and transmitting the skills and modes of acting that the productive praxis is carried out by. Thus, secondary artefacts are representations of the modes of acting in production; they are not merely pictures of objects or environments relevant to production, but “pictures” of modes of acting on and with these objects (Wartofsky 1973, p. 202).

To simplify things, we can say that primary artefacts are mediating production whereas secondary artefacts mediate communication and reproduction. Wartofsky constructs a graphical model explaining how perception is part of praxis. It locates perception in the feedback loop of human praxis where it is mediated, or conditioned by production and communication, the fundamental modes of praxis, and by the artefacts mediating praxis. In the division of praxis into production and communication, he refers to Aristotle’s distinction between making and doing, separating the making of things from the relation between people.

In addition to the inseparable “on-line” loop of praxis mediated by primary and secondary artefacts, Wartofsky suggests an “off-line” loop of imaginative construction. This loop is mediated by *tertiary artefacts*, which are abstracted from their direct representational function. Tertiary artefacts are in a genetic sense rooted in the productive praxis but do not depend on it in any direct manner. They constitute an autonomous zone of free creation of visions transcending the existing way of production, and they influence and change productive praxis by changing the established modes of perception.

¹¹ In making a similar point, Mogensen (1994) uses a notion of time adopted from Heidegger, stating that human endeavour is directed to the future based on the past, and that the present is the meeting between future and past. Wartofsky’s account is preferred in this thesis because it is far less cryptic and because it is based in dialectical materialism.

(Techno music changes our way of hearing so that we are able to discover the music of offset printing machines and truck horns).

Any artefact is, or is part of, a complex of primary, secondary, and tertiary artefacts. The hammer is a primary artefact for driving nails existing in a complex with (mostly informal) secondary artefacts representing praxis with hammers. The quality of the concrete hammer, particularly beautiful or lousy, the feeling of driving nails, the hammer as general metaphor for tool is, or induces, images with an amount of tertiary artefactness.

To sum up:

- Primary artefacts directly mediate productive praxis.
- Secondary artefacts are representations.
- Tertiary artefacts mediate imagination or conception of new motives/needs — radical alternatives to the well-known.

Chapter 5

A notion of design and design artefacts

The previous chapter outlined the theoretical preconditions for the construction of the notion of design and design artefacts introduced in this chapter. Activity theory is the backdrop on which everything is staged. The most important inputs from activity theory are the structure of activity and the double triangle figure, the concepts of mediation, contradiction, and development, as well as the notion of crystallisation. Wartofsky's analysis of perception and representation as outward action, as well as the concepts of primary, secondary, and tertiary artefacts are central for the notions introduced below. Finally, the sensitivity to heterogeneity as well as the concept of boundary objects supplied by Star are also central elements.

Design activity

Design is deliberate development of activity through the creation of new artefacts and their introduction into a given activity system — it is the (re-)creation of conditions for life. Design of computer artefacts is the development of the entire work setting, thus general notions of development and progress are important elements in understanding design of computer artefacts.

Analytically, design activity is when a designing subject shapes the design object by means of some design artefacts. The *design object* is the artefacts changed or created in design—the outcome which design activity is directed to. The design object is a part of the deliberately shaped environment, or conditions for human life: houses, cars, word-processors, and furniture. The *designing subject* is almost always a collective subject. Some of the members of this subject can be professional designers who design the conditions for the lives of somebody else, and others can be members of the praxis that the design object is intended to mediate. *Design artefacts* are artefacts that mediate the design activity; they are utilised but not consumed during the process. They serve as conditions or environment for the design process. Design artefacts can thus be opposed to materials. Examples of design artefacts are

programming languages, CASE-tools, specification standards, and systems development methods. A prototype can both be understood as a design artefact and as object of design. Design artefacts are mediating three main design functions: getting knowledge and understanding about what is designed, i.e. conception; communicating and co-operating during this process; and constructing the new artefacts forming the considered world. Often a design artefact is intended to support just one of these elements: an editor is used for writing code, a future workshop is used in order to understand the problem domain, memos are used for communication, and so on.

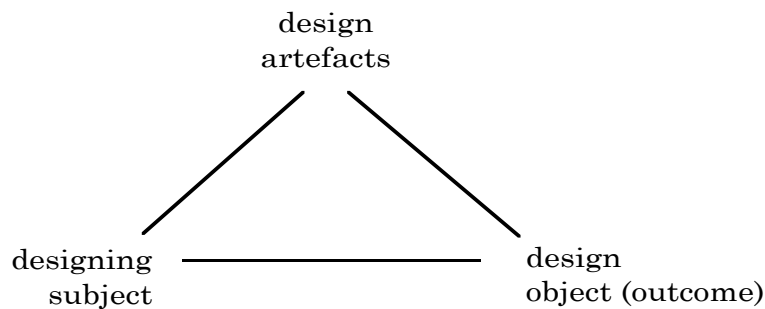


Figure 9: The relation between design subject, design object, and design artefact in naive activity theory terms.

Software is basically a moving target. From the point of view of activity theory, it is an obvious problem for any design discipline that the object of design will not take its final form before it becomes part of the object activity; furthermore the object activity is changed through the introduction of the new artefact. In the design of computer artefacts, the situation is complicated even more because computer artefacts are composed of symbols, or signs, which have meaning and makes sense in ways constituted in the situation of use. This unpredictability is a basic feature of design of computer artefacts and can only be remedied through iterative design processes involving representatives from the use domain.

The prototyping session (e.g. Bødker & Grønbaek 1996) can serve as a general metaphor for design. In idealised terms, prototyping is a process where one or more designers work together with one or more prospective users on developing new computer support for the users. Based on some sort of vision or an analysis of the users' needs the system developers build the first prototype. The prototype is tried out in work (like) situations with the users and gradually adjusted to fit the (renewed) praxis. Users and designers are fundamentally unable to understand each other, but during the prototyping

session they build a common understanding which, however, is only present as the final prototype.¹²

Thus, design is basically heteropraxial, i.e., involving heterogeneous groups of people with different backgrounds and different motivation for participating in the process.

In the framework of activity theory outlined above, design can be understood as the instrument producing activity in relation to the considered use activity, and the use activity can be understood as object activity for the design activity. This analysis, however, does not capture the intertwined character of the prototyping session proposed as a general metaphor for design. What tie the involved activities together in design are the involved artefacts; the design artefacts and the artefacts which are object of the design process. In the design situation these artefacts become boundary objects, constituting a boundary zone of design, where users and designers meet to change the world together but not necessarily understand each other (see figure 10).

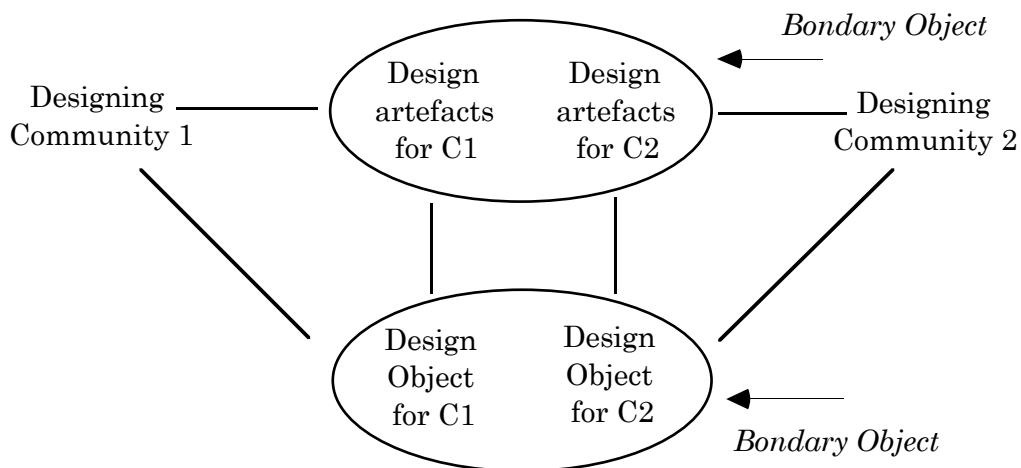


Figure 10: Heteropraxiality of design: by applying the notion of boundary object we get this picture. The figure is the composition of two triangles of (design) activity, the boundary objects and boundary artefacts of design constitute a boundary zone.

This analysis of design as being heteropraxial is primarily a descriptive one outlining general features of design. However, it also states the normative ideal of active involvement and codetermination by all involved parties as the basic form of unrestrained design; cooperative, or participatory, design becomes the ideal. Basically, participatory design occurs when users are ac-

¹² Actual prototyping always includes initial analysis of the considered activity system as a basis for the construction of the first prototype. The purpose in the present context is to point to fundamental characteristics of design in general, not to discuss rapid prototyping.

tive subjects in the design activity, which is opposed to the mere involvement of users as test objects. Thus, much of the so-called participatory design reported in the literature should rather be labelled extended user testing.

Three dimensions of design

Design activity can be analysed and decomposed in an infinite number of ways, according to perspective and desire to make yet another contribution. The identification of five systems development functions introduced by Mathiassen is just one example:

“[W]e postulate that change, decision, investigation, construction and communication are the necessary sub-functions which in some way or another have to be carried out throughout the systems development process and its sub-processes” (Mathiassen, 1981 p. 81, my translation).

In the specific context of Mathiassen’s work these sub-functions were useful, but in general they are logically inconsistent; e.g. change is a function at another level than the other sub-functions. However, the important achievement of Mathiassen in the present context is the notion of systems development functions being orthogonal to process. Loosely based on the notion of primary, secondary, and tertiary artefacts, the, it is possible to identify (or postulate) a more consistent set of three functional dimensions of design: construction, communication, and conception.

Construction

Construction is the productive relation between the designing subject and the object of design. System development is a process aiming at the construction of software and the environment for its use. In this way we are talking about system development as a programming process. The artefacts mediating the relation between designers and the design object seen as technical construction, are programming environments, CASE tools etc. The artefacts mediating the constructive aspects of design also mediate cooperation and conception. The use of a CASE tool makes the designers conceive the design object with the concepts supported by the tool, and will thus guide the process in certain directions. When looking at design as construction, the advantages of object-oriented programming languages are the possibilities of code reuse, the high degree of maintainability, and the stability of “code structure”.

Communication (or cooperation)

Communication is the representational relation between subjects cooperating in design. Design is a co-operative enterprise where different people with different professional backgrounds and different motives are engaged in cre-

ating something new. This is the reason why cooperation is essential in system development. The design artefacts mediate system development as cooperation, both as explicit means of cooperation (e.g. status reports and specifications) and as means for the sharing of experiences, insights, and visions about the design object. A prototype mediates cooperation when users through their exploration of the prototype yield knowledge about the use context to designers, and when designers express their new insights by way of continuously changing the prototype. When looking at design as communication, one advantage of object-oriented programming languages is that “physical modelling” is possible to understand for non-computer professionals. The communication dimension is related to the concept of articulation work referring to “the specific details of putting together tasks, task sequences, task clusters, and even the work done in aligning larger units such as sub projects, in order to accomplish the work” (Strauss et al. 1985, p. 175). We may say that articulation work is mediated by secondary artefacts.

Conception

Conception is the dialectical relation between the designing subjects and the historically developing activity. Design is conception of the considered praxis and of the new to come. Users and designers achieve new understanding of the existing world and rooted in this they conceive a new world that transcends the old. Design artefacts mediate learning and conception. A functional programming language for example facilitates the understanding of the design object as transformation of data streams. A prototype induces new visions and knowledge when it confronts the possible and the existing. Conception is about changing the existing *modus operandi*, both by identifying more suitable ways of realising the established activity and by developing entirely new motives. In the terms introduced later in the thesis, conception is both transformation and generation; it covers all degrees of radicality. Conception is both understanding (of the given) and imagination (of the better). When looking at design as conception, an advantage of object-oriented programming languages could be the ease of building explorative prototypes from existing objects.

A definition of design artefacts

The artefacts mediating design activity are called design artefacts and can be divided into two categories: global and local design artefacts. Global, or general, design artefacts are created independently of the actual project and exist before and after the project. Local design artefacts are specific sub-products of a project, and only applicable in that context, they are often representations - descriptions or models - mediating the transformation of

artefacts in the domain. Ehn and Sjögreen's (1991) notion of design artefacts is to some degree related to this concept of local design artefacts; however, their notion is limited to material representations produced during the design process as vehicles for the generation of visions. As a productive act, design is directly mediated by tools like compilers, editors, debuggers, and case tools. However, methods and theories are equally important as design artefacts.

Previously, the notion of human activity has been sketched, and the basic features of design has been laid out. Design has been described as a hetero-praxial activity aiming at the creation of new conditions for (day to day) activity. On this background it is possible to outline the main features of design artefacts.

Definition of design artefacts:

1: Design artefacts mediate the three functional dimensions of design (construction, communications, and conception).

a) design artefacts are, or belong to clusters of, primary, secondary, and tertiary artefacts.

2: Design artefacts have the double characters of mediating technical construction and representation; being tools and signs, having meaning and sense.

3: Design artefacts are boundary objects;

a) tying the activity systems involved in a project together,

b) tying different rooms of design and use together.

Mediating the three design dimensions

By definition design artefacts mediate design, thus they also mediate the three design functions. However, more interesting, any design artefact, no matter what its intended use is, to some extent mediate all three functional dimensions of design. E.g. a debugger mediates construction by providing efficient feedback when the programmer is coding; it mediates communication and cooperation by representing programs and bugs in a standardised way, thus facilitating programmers cooperative work on the program; it mediates conception by yielding alternative views on the actual program and serving as a way to learn about other programmers code, e.g. in standard libraries. In the same way a system description mediates construction by being a kind of very high level programming tool; it mediates communication and cooperation by fixating ideas to be shared and by being a vehicle for the division of labour; it mediates conception by inducing alternative, structured views into the problem in question. Mogensen (1992, 1994) has in detail explained how computer artefacts, both existing systems and prototypes, mediate concep-

tion by provoking praxis to become explicit. Basically, his idea is that by introducing new artefacts into a given praxis, limitations and possibilities become explicit. Design artefacts in general mediate communication and cooperation due to the secondary artefactness of the actual artefact or because it is clustered with secondary artefacts. Even gadgets which do not make sense will mediate communication because they can be pointed to and blamed for not making sense. Design artefacts in general mediate construction in the abstract sense that the outcome of design is the construction of a new computer application. More concretely any specification in design can be regarded as a very high level programming. In the sections about transformers and abductors (see below) it is discussed in detail how design artefacts are secondary artefacts and tertiary artefacts respectively.

The double character of design artefacts

The double character of design artefacts appears in relation to several aspects of design. Design artefacts have a double character of both mediating the technical construction of computer systems, which is an extremely formalised aspect, and mediating representational activity, which depends on a large degree of openness. This double character is a problem because the plasticity of secondary artefacts are obstructing or obstructed by the naturalism and formalisation of primary artefactness. The example below illustrates this feature of design artefacts.

Structured design (Yourdon 1982) is, from a technical perspective, a kind of very high level program construction tool, at the same time this tool serves as mediator of communication and conception. The applicability of elements from structured design in communication and conception with users depends on the designers' success in establishing common representations based on the tool. In 1990 we conducted a project based on a series of open ended, qualitative interviews with systems developers in a small in-house development organisation (Laursen et al. 1990). Focusing the interviews on communication and documentation in a particular project, we saw how output from a CASE-tool both facilitated and obstructed user-designer cooperation according to various circumstances. In one particular situation, a context diagram (poetically referred to as the "the sunflower") was successfully applied as a common frame of reference shared by the whole project group during the project. In an other situation the designers had no luck trying to apply diagrams from the same CASE-tool in communication with users in the project group. In both situations the CASE tool and its output had a double role, both mediating production of machine executable code and mediating communication and conception. In the first situation (the sunflower) the formalised features were weak enough to allow the formalised description to acquire other meanings, whereas in the other situation the model was so complicated that it was impossible to transcend just making sense of the

formal contents of the figure. The “sunflower” offered openings into a poetic world, whereas other diagrams only yielded frustration.

In a Vygotskian terminology, design artefacts are both *tools* (technical instruments) and *signs* (psychological instruments). This directs us to a wider use of the distinction between *meaning and sense*, not only in language but in relation to artefacts in general. Technical instruments have both meaning and sense. The meaning of a hammer is the assemblage of shaped iron and wood. The sense of the hammer is that it is used by carpenters for driving nails. The sense of the hammer is “crystallised” in secondary artefacts, mediating hammering praxis; e.g. anecdotes and sayings. The meaning of the “Sunflower” described above was that it was a (formalised) SA/SD context diagram. The sense of the “Sunflower” in the cooperation between designers and users was the context of the prospective computer system, and the way it would mediate parts of the relations between activities in the organisation. This has some degree of tertiary artefactness to it, because the poetic connotations of summer and nature may have mediated the emergence of entirely new ways of understanding the workplace.

Programming shares the same kind of doubleness. This is pointed out by Peter Naur (1985) who rejects the idea that programming can be seen as the mere production of machine executable code — isolated construction without elements of conception and cooperation. He explains that a theory about what the program does, and how it does it, is built simultaneously with the construction of the executable code. Thus, in the terminology introduced above, programming work has the double character of being both construction and conception. This theory cannot be written down or otherwise formalised, and is only accessible to the programmers working on the particular project. In Wartofsky’s terminology Naur’s “program theory” is a secondary artefact conserving the acquired knowledge and skills in working with the program. However, Naur bases his analysis on the individualist philosophy of Ryle, thus neglecting the societal/cultural aspect of representation. The point in the concept of secondary artefacts is that the way humans understand their surroundings (including programs) is culturally mediated, secondary artefacts not only conserve knowledge and skill among the individuals whose experience they are based on, but secondary artefacts also *transfer* these across a given culture, e.g. the programming profession. Furthermore, the empirical cases Naur uses in making the point about the impossibility of transferring “program theory” are ambiguous, and may be interpreted as pointing to the lacking ability of established design artefacts in transferring “program theory”. Thus, Wartofsky’s (1973) general statement that it is a basic feature of human praxis that experiences can be shared and transferred, cannot be rejected based on Naur’s programming as theory building idea.

Design artefacts are boundary objects

Design artefacts, global/general as well as local, are boundary objects; tying involved activity systems together, and tying different rooms of design and use together.

The heterogeneous activity systems contributing to design are tied together through their joint use of artefacts, and through their joint focus on the same object. Users and designers are driven by different motives and the object of design does not make sense in the same way for them. Design artefacts are boundary objects in the sense that they tie different praxes together, maintaining meaning across groups but making sense in different ways. When designers and users work together on a system specification, designers may perceive the specification as a model of the data flowing in the prospective system, whereas users may understand the same specification in terms of new ways of working in the organisation. In the same way, the double character of design artefacts described above also makes them boundary objects within one praxis.

Design artefacts do not only take different shapes or serve different purposes in different groups, they also take different functions within one group across time, during use and design, and in the different rooms of a specific group's praxis. Thus a system development method is a boundary object in the sense that it has one function in the project organisation's internal education prior to a project, and another function during the project. Working in accordance with the method means two different things in the two project rooms (see the example in the section on prescription versus praxis).

In the same way as design artefacts are boundary objects, the object of design, according to the prototyping as a general metaphor for design example, is a boundary object. Realising that design artefacts and the object of design are boundary objects, makes it possible to revise common ideas of shared understanding and interpretation in design. The crucial point is not for the designers to be able to understand and interpret the users correctly, but to supply design artefacts that can serve as boundary objects in mediating the design process.

The "sunflower" (described above) was a boundary object in the sense that designers perceived it as a computer system description whereas users perceived it as an image of the organisation, but it maintained stability across these different perceptions. In the Festival case the checklist was a boundary object in a number of ways (see the transformation of the festival checklist example below).

Chapter 6

Exploring the notion of design artefacts

In this chapter, the notion of design artefacts introduced in the previous chapter will be further explored. Firstly, to demonstrate the scope of the design artefact concept, the relation between theory and design is discussed by viewing theory as design artefacts. Secondly, the tension between principles and praxis in systems development is discussed as a special case of the general doubleness in design artefacts of meaning and sense. Thirdly, transformers are introduced as a norm for the secondary artefactness of design artefacts, i.e. the use of representations in design. Finally, the notion of abductors is introduced as a vehicle for the discussion of the tertiary artefactness of design artefacts and artefacts in use.

Theories as design artefacts

Scientific theories form an important class of representations, which are formalised and pretend to be independent of concrete praxis. Because of the very formalised quality of some theories it is possible to use them directly as mediators of productive praxis. Thus, it makes sense to explore theories as design artefacts.

Orthodox cognitive science based human-computer interaction research believed that design tools could be built by making approximate versions of the general theories (Card et al. 1983). Although the rigid hard science vision of Newell and Card (1985) is not shared by everybody in the human-computer interaction community, the ideals of the natural sciences are widely accepted. Universal disinterested knowledge is seen as the only alternative to unsystematic design by rules of thumb and *ad hoc* procedures. This general lack of theories that are pragmatically rooted in praxis has been destructive for the entire field of human-computer interaction (Bertelsen 1993).

Theories can play different roles in design of computer artefacts; from world views, guiding the designer and helping him assess the situations and keep

the goals in mind, to tools mediating the achievement of specific results. In (Bertelsen, 1994b) the notion of theories as design artefacts is introduced through the example of Fitts' law.

Fitts' law is one of the most cited results of experimental psychology in the field of human-computer interaction. Every year papers based on this law are presented. Thus, it is an obvious case to examine in trying to understand the role of theory. Based on mathematical communication theory (Shannon & Weaver 1949) Fitts' law (Fitts 1954) states that there is a certain relation between how fast the human hand can be moved and the length and precision of the movement, and further that this relation is congruent with an ideal mathematical communication channel. In the classical human-computer interaction "Fitts' law study" Card, English, and Burr (1978) compared the performance of various pointing devices in a text selection task. These results were the basis for incorporating law into "the model human processor" by Card, Moran and Newell (1983).

Three distinct "design artefact roles" played by Fitts' law can be identified based on the literature (Bertelsen op cit.). These are: basic world view, tool for calculation, and source of inspiration. In the human-computer interaction psychology by Card et al. (1983) Fitts' law was part of the *world view*. Basically, Fitts' law assumes that human behaviour can be understood through additive models, i.e. by dividing it into behavioural atoms which can be analysed in isolation and then sum up the findings. Thus the world view promoted by the use of Fitts' law is that human beings are computing machinery. Landauer (1991) reports on the use of Fitts' law as a *tool* for specific optimisation of a screen layout, which does not necessarily imply cognitive science as a main perspective on human-computer interaction. Gillan et al. (1990) have used Fitts' law as a *metaphor for research* on specific aspects of human behaviour with a mouse. In design this is equal to the use of the Fitts' effect as a thinking tool (Gediga and Wolff 1989).

The borders between these three roles are not clear-cut. By thinking about interface problems in Fitts' law terms (the metaphor role), other views are excluded and Fitts' law draws the main perspective on the interface towards time performance concerns. As a tool for calculation Fitts' law is a primary artefact, just as a microscope. Of course, the things we are able to see by using Fitts' law are irrelevant for design unless the calculations can be related to the use situation. As a metaphor, Fitts' law is a secondary design artefact. As part of the basic world view, Fitts' law promotes the idea that human-computer interaction can be decomposed and subjected to universally valid laboratory experiments. In the later role, Fitts' law and the entire cognitive science framework is a secondary artefact.

These three roles of Fitts' law as a design artefact cannot be generalised, but they span a spectrum of ways in which theories mediate design. The two extremes in this continuum of roles are:

- tool: goal directed, “neutral”, and
- world view: value based, motive oriented.

Activity theory takes the tool role when analysing empirical data of interaction with a specific computer artefact in terms of focus shifts (Bødker 1991). If designers adopt the framework of activity theory as their way of understanding technology, i.e. mediating human activity which is culturally mediated etc., then activity theory takes the world view role. In between these roles the double triangle figure can serve as a vehicle for communication, and in the human-computer interaction community in general activity has become a metaphor for seeing that there is more to designing computer artefacts than time and motion optimising (see Blumenthal 1995, for an example of this use of activity theory).

In a logical positivist perspective there is no difference between the two roles of theory. Theories are universal and can be applied independently of context and purpose; they are always true and therefore need no direction related to motive and values. If, however, we reject the idea of a universal and value-free science also the most specific tool-like applications of theory, are carrying a certain world view. This is no surprise; it is a corollary of the relation between perception and action as analysed by Wartofsky.

The world view role is not only played by explicit theories. In any design situation, a set of basic assumptions are always playing an active role in guiding the process and determining the outcome no matter if they are explicit or not.

More than thirty years ago Meister and Farr (1967) showed that engineers were disinclined to use human factors information that was not presented as graphs or tables. In general, engineers and other designers can be expected to be pragmatic people, interested in getting the job done rather than engaging in metaphysical questions about the fundamental arrangement of “the world”. Thus, an explicit world view is likely to be absent in many design situations leaving the arena for either mechanical materialism or whatever world view embedded in the applied “tools” to govern the design process. This can cause problems in using reductionist based approaches like GOMS (Card et al. 1983). In general, GOMS is not a harmful approach if it is utilised as one tool among others, and if it is subsumed under a basic understanding of “the world” that acknowledges human beings as human beings. However, in a situation without a strong ruling world view GOMS will act as a Trojan horse for the basic assumptions of the very mechanistic cognitive science of Card et al. (1983).

Theories as design artefacts, as exemplified by Fitts’ law, conform to the general notion of design artefacts developed in the previous chapter.

Theories mediate the three functional dimensions of design by playing the roles in the continuum from tools to world views, being primary, secondary

and tertiary artefacts. Theories explicate the doubleness of design artefacts of mediating construction and representation; as in the case with Fitts' law both mediating technical optimisation of user interface layout, and representing specific understandings of the interface design problem. Theories, like other design artefacts, are boundary objects, most prominently by tying the production of “universal facts” in research together with design. The notion of theories as design artefacts contributes to a pragmatic philosophy of science by providing a criterion for the assessment of theory. Thus, ideally, the theoretical framework introduced in this thesis should be assessed as a design artefact.

Principles versus praxis

The recurring conflict between principles and praxis in systems development is an instance of the general tension between meaning and sense in design artefacts. The two examples below are related to methods; in the first case a method book, in the other the object-oriented “method” embracing the Mjølner BETA system together with a specific tool.

In most cases a specific design artefact is intended to be used in a specific way, and just as often the actual way the design artefact mediates design is very different from this intention. In most cases, the relation between the principles of intended use and the actual praxis with the artefact is far from trivial. The mismatch is most obvious in the case of design methods, which is illustrated by the below excerpt (reconstructed from interview notes) from one of the interviews with designers of embedded software at a Danish machine factory.

Olav: How do you do when you specify the software for the pumps?

Designer: well, according to the SPU method [lengthy summary of method book].

Olav: OK. The pump we looked at in the front room, the one you designed the software for, how did you make the specification for that?

Designer: With that particular pump the situation was a little unusual, so we did not go through the steps of the SPU method.

Olav: Then, if we take the project you are currently involved with. How is specification done there?

Designer: The current project is not a regular design project, because we are also developing a general framework.

The embedded software department at the machine factory had used some effort in implementing the SPU method, but the interviewed designer had

never gone through the steps, defined in the method book, in a project. This is in accordance with findings of Button & Sharrock (1994) who studied the use of design methods in a development project and found a tension between methodologists' ambition (clarity, coherence and closure) and the indeterminate character of actual design work. This mismatch between the prescriptions of the method and the actual design work can be understood in terms of the doubleness of meaning and sense of the method. The meaning of the method being the literal prescription of how to proceed through specified steps, filling in certain forms etc.; the sense of the method being the actual use of it, the sparse use of selected techniques combined with common sense, etc. However, when the distance between this "meaning" and "sense" becomes too long, the meaning-sense relation degenerates to a contradiction between principles and praxis. Argyris & Schön (1978) write about the tension between espoused theory and theory in use in a somewhat similar manner.

In our study of the object-oriented source level debugger, Valhalla, we saw similar problems (see the chapter "Two Empirical Cases"). The tool was based on strict object-oriented principles, whereas the users to a great extent worked with programs as text. The tension between principles of object-orientation and actual programming praxis was most clearly expressed during the re-design workshop.

Valhalla was intended to be an *object-oriented* debugger in contrast to traditional source level debuggers. It is a part of the Mjølner BETA system, which is built upon a conceptual framework emphasising that programming is modelling, but also focusing on static aspects rather than the dynamic of execution. However, in debugging it is important to be able to monitor the dynamics of execution — the debugger was conforming to principles rather than supporting praxis.

In discussing the navigational problems in the use of Valhalla, the principles versus praxis tension turned up again. The designers discussed whether programmers had, or should be allowed to have, a textual understanding of programs. The claim from one of the designers that "if you are working on the text then you are working in a procedural [non object-oriented /ob.] manner" was later moderated by another designer acknowledging that "the program text is a suitable means of navigation". But, it was emphasised that "the true object-oriented programmer" is not working on code-text but on the structure of the program, i.e. hierarchies, patterns, objects. The view on the textual representation as well as the view on specific executions are only means for viewing the structure.

The principles of object-orientation prevented the designers, from acknowledging the central role of textual representations in programming praxis, including their own praxis. Valhalla was primarily intended as a part of the

hidden curriculum for the re-education of programmers to do true object-oriented programming.

The designers were unsure if anything was gained from the object-oriented debugging approach. The discussion of their own use of Valhalla, had forced them into explicating their own debugging praxis. To meet the demands of their own praxis their debugger had to include support for more “traditional” debugging, e.g. tracing of variables, this was agreed to be featured in the next version along with object-oriented features. This was not considered a setback for the vision of object-oriented programming activity, rather, according to one of the designers, programmers could start using the tracing facilities and then gradually approach the new “religion”. In terms of the expansive cycle, the new instrument was not consolidated in a new object-oriented activity. The praxis of old time programmers to hand wire print statements was surviving the vision of the new object-oriented programming activity. This is not an instance of too little power in introducing the new, but rather due to a too limited scope of the vision. However, the introduction and consequent use of an object-oriented programming language into a programming praxis is no guarantee for the successful transition of the programming activity to be object-oriented. This is realised in the Mjølner community (Knudsen (ed.) 1994) and is the reason for the semi religious talk about structure over text, during the workshop.

The examples above point to a general contradiction between prescription and praxis. The general discrepancy in design artefacts between what they prescribe and the praxis they induce can in some situations be understood as a contradiction between the design artefact producing activity and design praxis. This is the situation in the case of the Mjølner programming environment (here Valhalla and Sif). The tool designers think that programmers need to understand their own work as constructing object-oriented structure, whereas the programmers, even when they are intimately familiar with the structure not text idea, continue to write program text. In this case, the instrument producing activity is also representing the more advanced activity. The funny twist in the case of the Mjølner environment is that the same persons are subjects in the instrument producing activity introducing new principles, and in the programming praxis rejecting to adopt these principles. In other situations, the discrepancy between principles and praxis can be understood as a result of the relation between meaning and sense in design artefacts. In the case of the designer of embedded software, the meaning of the SPU method was the prescriptions for design printed in the method book, whereas the sense of the method was the way it actually influenced work in concrete design projects. The interviews did, however, not shed sufficient light on how the method influenced praxis, but only that the designer had a bad conscience about not obeying the prescriptions in full detail. Methods, or method books, are secondary artefacts to the extent to which their prescriptions are rooted in praxis. But, according to Wartofsky,

secondary artefacts are not depending on naturalism. Some method books are based on fiction, war stories from other fields and the like, which in relation to the purpose of a method book has a character of tertiary artefactness. We cannot expect methods to prescribe praxis in a direct manner. Instead, the prescriptions embedded or expressed in design artefacts are resources; they do not determine process and product; they are seeds that can develop in concrete praxis in varying ways depending on specific circumstances.

Transformators the secondary artefactness

The secondary artefactness of design artefacts can be understood in terms of transformators¹³ and transformation. Transformators being the design artefacts mediating the creation of representations which can be regarded transformed versions of other artefacts. Despite that this thesis generally focuses on artefacts these concepts are introduced focusing on process. Transformation is a process oriented concept focusing on the connection between development in use and development by design. The concept is evolutionary, emphasising tradition over transcendence. While this focus on minimising the damage caused by a too abrupt introduction of too much new technology is shared by many systems development methods, it must be stressed that transcending the limitations of established praxis should be an equally important purpose of such methods.

Transformation connects use and design by turning artefacts mediating use praxis into local design artefacts, and back again. The idea is an outcome of the Music Festival project (see example below). The computer support for pre-production work, built in the Festival project can be understood as a new version of the paper based pre-production checklist. Furthermore *post hoc* analysis of the project (Bertelsen 1996b) showed that the lack of success of this new computer artefact can be attributed to lacking possibility of grounding the new artefact in praxis during the development process. The concept of transformation is consistent with the experience and techniques developed in the UTOPIA project and onwards (Bødker et al. 1987, Greenbaum & Kyng 1991).

In general, activity with one generation of an artefact is crystallised into the next generation (see above), thus the history of a praxis can be discovered by studying the history of the mediating artefacts. The idea of transformation is to base design on the historical development of the praxis by deliberately

¹³ The back-formation of the word transformator from transformation is used over transformer to emphasise the specific meaning of concept in the present context.

continuing the crystallisation of activity into artefacts. In the methodological cycle of developmental work research (Engeström 1987), this crystallisation of activity into artefacts is analysed in the object historical analysis of the second phase. The difference is that the object historical analysis is a technique for establishing background knowledge for the development process whereas design as transformation of artefacts concretely brings “the object history” into design.

In design by transformation, central artefacts from the considered praxis are taken into the zone of design. This depends on the boundary object capability of these artefacts, maintaining their identity across use and design. When the central artefacts have been brought (successfully) into design, they are gradually changed while they are mediating simulated praxis. By confronting the artefacts with praxis throughout the process it is ensured that the subsequent transformed versions make sense in praxis. When the transformation of the central artefacts is completed and they have taken their new form they are brought back into use.

The process of bringing central artefacts from praxis into design is a representation process; for this to be useful these artefacts must be boundary objects, which maintain stability across representation. When artefacts are transformed by design they exist both in their original form within the domain of use, and within the domain of design likely as representations. Thus artefacts being transformed are boundary objects in the sense that they exist in different activity systems, and tie these together across groups, across phases of praxis, and across the transition between use and design. The crucial aspect of design by transformation is that the transformed artefacts need to be robust enough to maintain their identity and still make sense in all the involved rooms of design and use. The main practical achievement of the transformation idea may be that it contributes to an increased sensitivity among designers to this boundary objectness.

The general concept of transformation is illustrated in figure 11. The big arrow is the central artefacts transformed over time, the big ovals are the main rooms in which transformation take place. The small ovals show that the transformed artefact is confronted with actual or simulated praxis throughout the process. The horizontal arrows at the right side indicate important general design artefacts. The partly overlapping circles in the last “stage” indicate that the transformed artefact is spread to a wider range of related activities.

In terms of Engeström’s (1987) cycle of expansion, the big ovals on the design side of the transformation model may correspond to the “modelling of instruments” phase, and the small ovals may correspond to application and generalisation. Thus, transformation involves several cycles, which are likely to be restricted by the too tight connection to established praxis throughout the process.

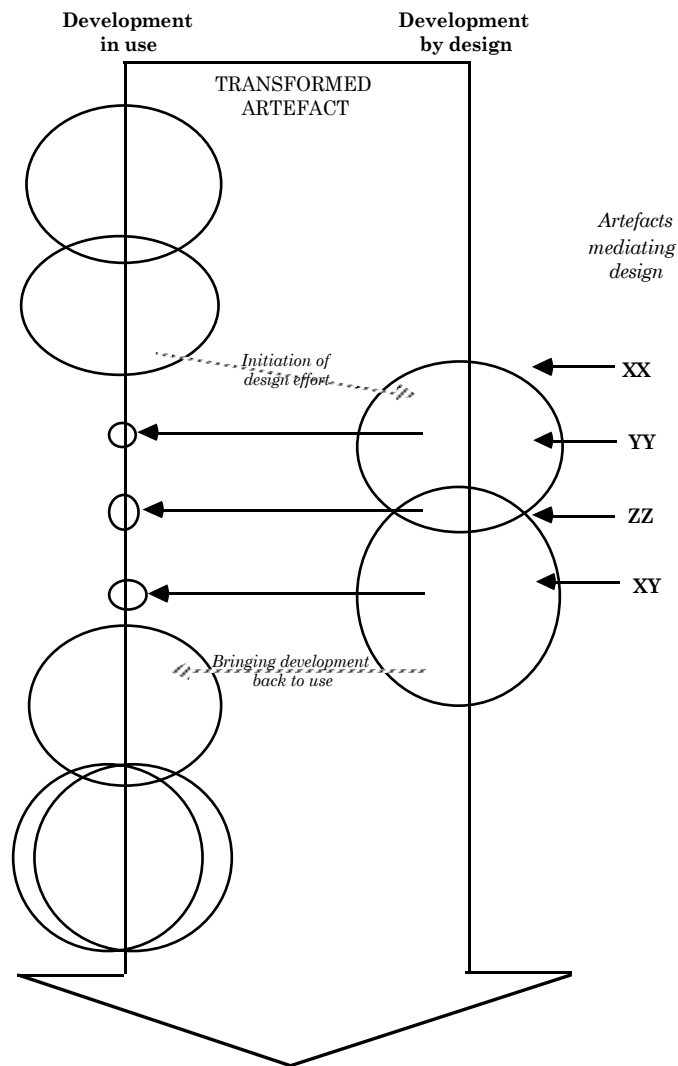


Figure 11: Design as transformation.

The transformation approach is in some sense a conservative one emphasising continuity over the development of radically new needs. In this sense, it is in line with the traditional approaches to office automation; and even bares morphological resemblance with structured analysis (Yourdon 1982), see figure 12. However, the differences between the transformation idea and traditional office automation are more prominent than the similarities. Traditional office automation is system-oriented, i.e. it understands design as optimisation of the input-output relation of a system with an external owner. While the purpose of office automation is to replace people by technology the transformation approach is praxis-oriented, focusing on the enhancement of mediated activity. The purpose of the transformation approach is to ensure that the outcome of the design process makes sense *in praxis*.

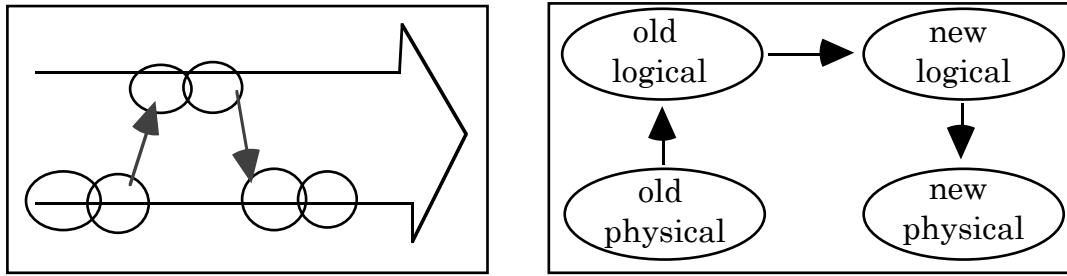


Figure 12: The morphological similarity between transformation and structured analysis.

Object-oriented modelling and transformation

The forefront of current system development methods are based on the *object-oriented* paradigm, here referred to as physical modelling. The physical modelling paradigm and its relation to the transformation approach is described in some detail in (Bertelsen 1996a).

Traditionally, the advantage of physical modelling has been ascribed to the naturalism of such descriptions (e.g. Holbæk-Hansen et al., 1975, Madsen et al. 1993, Coad and Yourdon 1990). Physical modelling is convenient in modelling very complex, technical systems, including the computer itself, but it is unclear how physical modelling facilitates or prohibits active involvement of users in design¹⁴. However, by way of the concept of transformation it is possible to justify this idea.

According to Wartofsky's notion of secondary artefacts, representations are not one to one naturalist pictures of interesting parts of the world, but rather containers for knowledge and skill related to outward action. Representations are important components embedded in any praxis, mediating the ongoing production and reproduction. Thus, system development description and modelling artefacts should not be evaluated with "naturalist" criteria, but rather by looking at the actual design praxis made possible by the formalism.

Madsen et al. (1993) refer to the paper-based Norwegian train reservation system (Sørgaard 1988), and medical records to point out that physical models are not only parts of computerised systems, but also important parts of human praxis in general. Physical models are secondary artefacts.

¹⁴ Colleagues outside the DEVISE project have asked why I am interested in the relation between object oriented systems development and active user involvement in design. The answer is that I wasn't in the first place, but it was a basic assumption of the DEVISE project that object oriented programming was good for user involvement (see Bødker et al. 1988).

Thus, the advantage of physical modelling in designing computer artefacts together with non-computer professional users is that physical models are integral parts of most praxes, and that physical and formalised objects can be modelled and transformed as physical models in a way that makes sense for the involved parties.

By understanding physical modelling based design as transformation of artefacts it is possible to transcend the naive naturalism expressed in some object-oriented approaches, (e.g. Coad & Yourdon 1990). Thus, the reformulation of examples, like the Norwegian train (Sørgaard 1988), of object-oriented design in terms of crystallisation and transformation processes, validates the use of physical modelling in design together with users.

Transformation of the festival checklist — an example

To exemplify the transformation idea the design process in the Music Festival project can be analysed as transformation of the checklist. The transformation of the checklist is illustrated in figure 13.

The original checklist as it was made by the Green stage group was a crystallisation of festival work. They were doing the same things every year when they received information from pre-production, and when the artists arrived at the festival. They made notes on sheets of paper and these notes gradually became more standardised and, in the end, the pre-printed checklist was made. Thus knowledge about how to receive the artists and what to look out for was crystallised into a new artefact the pre-printed checklist.

When the Sound and Light group took over the checklist, it was transferred from the local context of Green stage group, to the rest of the organisation. In this form the checklist served a broader range of functions; it became a planning tool and a media for information exchange. The checklist became a boundary object.

In the design process, the checklist was initially transformed into a database sketch by the Sound and Light group; the checklist became a local design artefact. For Sound and Light it was an incarnation of a technological vision, and it was a representation of pre-production work, and of how this work supports production on stages. For the researchers the database sketch was first a too narrow technical vision, but later, during the last part of the workshop, it became the specification for the prototype. The object-oriented descriptions made by the researchers were transformations of the checklist in the sense that they were a step in the definition of the relational tables in the prototype. For the researchers the prototype was the new checklist, with which the Sound and Light group could do everything they previously did with the paper checklist. However, an important aspect of the paper-based checklist was that it was handed over to, and used by the stage groups. The way this was done with the prototype was that the information

in the database was printed out on paper and attached to the band files; thus, the reincarnated checklist returned to the stage groups. Unfortunately, these printouts did not make sense as a checklist for the stage groups, they contained all the information collected during pre-production but lacked all the empty fields that were the original feature that mediated stage work. The grey cross on the checklist transformation figure (figure 13) indicates that it was not possible, for political reasons, to confront the design proposals with actual, or simulated, festival work.

The checklist existed both in the domain and in the design process (database sketch). It was a boundary object in the sense that it existed in different activity systems, and tied these together; across different activity groups; across the different, incommensurable stages of the individual groups cycle of the year; and across the transition from use to design. However, the checklist was not robust enough to be carried from the Festival, to the researchers and back into the Festival. In the transformation of the checklist from paper-based checklist into computer-based pre-production support it lost its “checklist-ness”. The researchers saw the checklist as a medium for the transfer of the information gathered together during pre-production, for them it was an incomplete, non-computer version of the future computer system; but for the people working at the stages, the checklist had its main function as a tool for the preparation and reception of the individual artists.

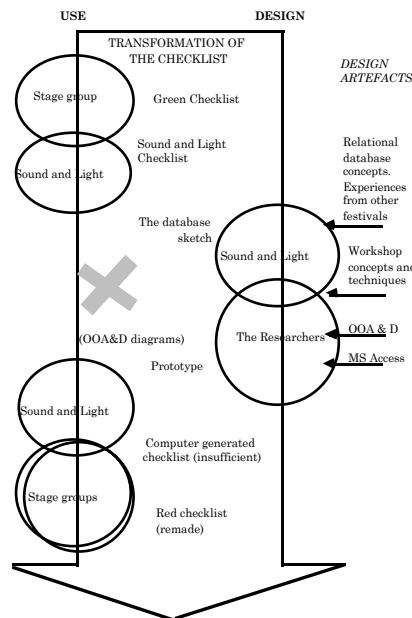


Figure 13: The checklist history, an example of an unsuccessful transformation process.

Abductors the tertiary artefactness

In this section, the tertiary artefactness of design artefacts, as well as the tertiary artefactness of artefacts in use, is loosely unfolded by way of the notion of abductors¹⁵.

Whereas the notion of transformation introduced in the previous section is directed to the secondary artefact aspect of design artefacts, focusing on maintaining praxis across the introduction of new technology, the notion of abductors is directed to the tertiary artefactness of design artefacts emphasising the creation of unexpected new motives and new modes of action. In contrast to transformation, dragging use into design, abductors in some sense infuse design into use. Abductors are the tertiary artefact aspect of the artefact clusters mediating design as well as use. In the most radical sense the notion of tertiary artefacts imply that they cannot be planned for as part of any purposeful act. However, it makes sense to put instruments mediating the creation of unexpected solutions to recognised problems into this category also. Thus, two sub-categories of abductors can be defined according to the presence or absence of a recognised need state, and the subsequent entry into deliberate development. Abductors in the radical sense, are e.g. abstract paintings, reshaping our ways of perceiving the world, whereas abductors in a plain sense could be future workshops (Jungk & Müllert 1987) mediating the unexpected solution to a need state.

Whereas Engeström (1987 and 1996) describes creation of the new as being part of progressive projects, the tertiary artefacts as introduced by Wartofsky reside in an off-line loop de-emphasising the project aspect. The separation of goals from motives (Leontjev 1981) caused a separation of tools as purpose by themselves, whereby secondary artefacts emerged. As this separation is intensified in modernity, a stationary state of constant innovation arises (Marx & Engels 1848); the dynamics of innovation and imagination become separated autonomous motive, thus giving rise to tertiary artefacts. In the form of aesthetic expressions tertiary artefacts are created by individuals as a result of individual genius (explainable as the result of the experience of contingency in modernity, e.g. Baudelaire), but if such forms of creating are going to play a role in design they have to be “socialised” and objectified.

¹⁵ In some of the papers of this thesis, abductors are called generators, but to avoid confusion with the “generators of secondary artefacts” referred to in other papers, I have chosen another term. I do not intend to indicate any relation to Pierce’s use of the concept abduction. Other terms could have had their own drawbacks; expanders would have suggested mediation of expansive learning in a need state, transcendents, would have implied too much dialectics and an unintended relation to Pelle Ehn’s tension between tradition and transcendence.

In meeting a recognised need situation, the instruments described by Engeström (1987), together with established techniques in cooperative design (see, e.g. Greenbaum & Kyng 1991), form a broad collection of plain abductors. Engeström (1987) emphasises the role of theoretical models in expansive learning; similarly, models in systems development play a strong role, not only as very high level programming, and as transformations of artefacts from use praxis, but also as abductors, generating conceptions into the unexpected new. The modularity and recombineability of object-oriented models may make them particularly important as plain abductors. However, the use of object-oriented design artefacts as abductors may as well restrict the process by enforcing a too narrow focus on information and computer technology per se.

With respect to radical abductors, the situation is more open. The horizontal forms of development described by Engeström (1996) rely less on a recognised need state, thus, implicitly announcing an agenda for future development of activity theory addressing the issue of radical abductors.

In use, radical abductors are rooted in tertiary artefactness in the general sense of changing established modes of perception, it is a matter of aesthetics, and it is related to the suitability of the specific computer artefacts as re-mediators (see above). An important point here could be to make less “well-planned” user interfaces, allowing for poetic openings into contingency and imagination. Being distracted at a lecture, you might look through the windows, observing the forest of TV-antennas on the roof tops; in your imagination the TV-antennas become a jazz orchestra; when focusing back on the lecture you may understand organisational games, the topic of the lecture, in a new way as a jazz band. In a computer interface there is no functional point in seeing the “TV-antennas”, so they are not visible. The obvious difficulty is that the designer (in principle) has full control of the 1100 x 800 pixels on the screen to exclude anything that does not contribute to “getting the job done”, and error messages from the basic system software coming through to the user is generally considered a very bad idea with good reason. Furthermore, abductors in use points to aspects of the artefacts explicating current modus operandi, by functioning as Marxo-Freudian tools for creating consciousness of existing limitations in order to transcend these. This also points to the issue of enabling re-mediation by avoiding over specialisation (Jones 1988). Clearly, facilitating development in use in this way is very different from embedding a curriculum for the development of transparent interaction, into the interface (Bardram & Bertelsen 1995).

The notion of radical abductors in design is close to being a contradiction in terms, because design is a purposeful endeavour. In contrast, radical abductors would be mediating purposeless systems development activities. When John Whiteside, back in the eighties, sent an entire department at DEC on leave to read Heidegger’s “*Sein und Zeit*”, that book may have served as an

abductor when the department came back. Suggestions for possible abductors could include “wild pictures” unrelated to any possible problem, or techniques based on Fluxus or Dada. However, when such elements are drawn into the design process they may end up as plain abductors, nicely addressing “the problem” in question. Thus, in general, radical abductors are likely to reside, and stay, outside the development process, belonging to, e.g. general education, hobby, or art.

Conclusions of the thesis

“not either one,
not both combined,
but both alone,
connected and transcended.”
–Engeström

A materialist and dialectical framework for understanding and influencing the field of use and design of computer artefacts has been introduced. Materialist because it insists on the reality of the material world and the material character of mental phenomena; dialectical because it rejects the idea that human life is a mechanical product of its material basis (including the physiological structure of the brain); the mental and the material determining each other in a dialectical relation. Activity theory has been pointed to as a possible, although not unproblematic, basic framework for systems development research; integrating the relevant and necessary aspects involved in designing computer artefacts (human-computer interaction, design, organisation, programming, cooperation, etc.), not necessarily operational at a detailed technical level, but as a general “world view” under which the sub-fields can be integrated.

The concept of design artefacts has been introduced as a unifying perspective on systems development, emphasising material mediation in design. The concept is based on a dialectical materialist approach comprising activity theory as a general perspective (mainly Engeström), and specifically the notion of primary, secondary and tertiary artefacts (Wartofsky). The dialectical materialist background has been supplemented with the notion of boundary objects (Star), as mediators in boundary zones. The argument has been based on the tenet of activity theory that human praxis is mediated by artefacts and is continually changing in the process of socio cultural development; and further that the history of praxis is crystallised into artefacts. Design has been pictured as a zone where heterogeneous praxes meet to change a given praxis through the construction and introduction of new artefacts, mediated by design artefacts making different sense to the various praxes (boundary objects). As special instances of design artefacts, transformers, mediating design as a transformation process emulating the process of development in use, and abductors, mediating the development of new motives, have been introduced.

Design artefacts

The idea in activity theory that human endeavour is mediated by culturally developing artefacts permeates the entire contribution. In understanding design and in understanding the relation between research and design, this tenet is expressed in the notion of design artefacts, which is the central concept introduced in the thesis. With reference to Wartofsky's terminology, it has been argued that design artefacts are clusters of primary secondary and tertiary artefacts (or they are part of such), implying that they mediate the direct production of the new computer artefact, that they are representing the considered work praxis as well as design work and, finally, that they take part in a non-productive off-line loop of free imagination. With reference to the concept of boundary objects and the concept of zone discussed by Engeström (1996) and others, it has been argued that design artefacts mediate design in a boundary zone, where heterogeneous praxes meet to create the new. With reference to the distinction between primary and secondary artefacts and the general tension in language between meaning and sense (Vygotsky), it has been argued that design artefacts have a precarious double character often emerging as a conflict between principles and praxis. The concept of design artefacts and mediation is a unifying concept in the sense that it is possible to understand all outcomes of systems development research, and computer science¹⁶ in general, as design artefacts, and to appreciate the value of these outcomes according to how they mediate design.

Development in use and use in design

Connecting use and design is an old idea, exemplified by the involvement of users in design, and by the concept of tailorable systems. To maintain the focus on use quality as a product of the development of use itself, the important concept of tailorability (Trigg et al. 1987, Henderson & Kyng 1991, Mørch 1997) has been left out of the discussion. As indicated in the discussion of support for the development of transparent interaction (Bardram & Bertelsen 1995), computer artefacts are changing during use without being altered technically (e.g. by tailoring). With reference to the notion of crystallisation of activity into successive generations of artefacts, it has been argued that the notion of design as the transformation of artefacts from the domain of use, is connecting design and use, ensuring that the new computer artefact makes sense in the considered praxis. Transformation of artefacts establishes a boundary zone of use and design, the transformed artefacts being boundary objects in the double sense of both mediating across heterogeneous communities and across the use-design border. It has been argued that artefacts maintain identity across transformation, abstractly as they

¹⁶Computer science, not only meaning the sub-disciplines which Naur calls Datalogy, but the whole spectrum of fields of research dealing with use and design of computer artefacts.

continue to make sense in the same way for the involved praxes, and because representations in design in this situation are the secondary artefacts maintaining praxis, made explicit. As a particular result, it has been argued that understanding representations in design in terms of transformation yields a solution to the recurring referent system problem in object-oriented methods.

Radically pragmatic philosophy of science

Based on the notion of design artefacts, a radically pragmatic philosophy of science has been suggested. Pragmatic, because the validity of a theory is appreciated based on its mediation of design activity (or other praxis). Radically pragmatic, because validity is not appraised based on the random preferences of detached individuals, but based on the reality of concrete societal praxis at a specific point on the trajectory of cultural development; thus, neither relativism nor utilitarianism. This normative approach to the study of theory and history of disciplines dealing with use and design of computer artefacts has yielded a solution to the difficulties experienced earlier (Bertelsen 1993) in trying to reject the orthodox cognitivist engineering psychology of human-computer interaction (Card et al. 1983). However, it has also induced enhanced sensitivity to the motives and achievements of these early contributions. The radical pragmatic philosophy of science makes it possible to both maintain earlier contributions and to destroy these in the creation of a dialectical materialist basis for design, thus formatting the basis of systems development research so that it has both an exclusive world view and is inclusive in the adaptation of earlier achievements.

“The new”

Proponents of activity theory have fanatically claimed that development and innovation are strictly collective achievements. Thus, Kuutti (1989) concludes that there is no need for the individualist concept of creativity, because all relevant aspects of development can be systematically understood in terms of expansive learning. By understanding development as being the collective answer to a need state, such positions miss that innovation in late modernity is becoming separated from production. It is essential to be able to comprehend, at least partially, the exceptional creativity of individuals, not in idealist terms but as a materialist theory of genius. Wartofsky's concept of tertiary artefacts provides such an understanding of individualist creativity as a material phenomenon in the off-line loop. Thus, tertiary artefacts is a basic concept in understanding creativity and innovation as a material phenomenon, in a way that transcends the limitations of the historically deterministic activity theory. Furthermore, by bridging the gap between individual inspiration and collective achievement, the notion of ter-

tiary artefacts will play an important role in maturing a dialectical materialist notion of horizontal development that is evading relativism.

Discussion of the contribution

The strength of the concept of design artefacts is that it provides a uniform perspective on the object and praxis of systems development research, the downside is that such perspectives in general tend to bring about tunnel vision, ignoring important aspects of (in this case) systems development, and systems development research.

The two important issues of project management and power struggle, seem to be absent from the framework introduced in the thesis. These two issues have, however, been the main focus of several earlier contributions to critical systems development research; power struggle was the main focus of the early Scandinavian projects, NJMF, DEMOS, DUE (see Ehn & Kyng, 1987, Bansler 1987), and project management was thoroughly dealt with in the MARS project (Andersen, et al. 1990). These issues are indispensable aspects of systems development research, but can as separate themes be regarded external to a conceptual framework for design. However, project management and power struggle are not excluded from the framework introduced in the thesis; power struggle is a basic dynamic force in the activity theory framework, understood as part of the primary contradiction expressed in terms of the contradiction between capital and labour; and project management is part of the communicational dimension of the introduced notion of design.

Obviously, process-oriented approaches yield important insights; however, the design artefact concept, introduced in this thesis, promotes a focus on the mediation of design praxis, rather than on principles and intentions. Tools are crystallised activity, secondary artefacts represent process, world views are carriers of conflict between interested parties. Thus, power, process and politics are profoundly embedded in the basis of the introduced concept of design artefacts.

Future work

The focus of the thesis has been on the development of a set of basic concepts for systems development research and praxis, based on a sparse amount of empirical material. Thus, an obvious direction for future research is to systematically use the introduced concepts in studies of development project, and in the identification and analysis of concrete classes of design artefacts. Such a direction could be taken within Centre for Object Technology.

New insights into the history and philosophy of systems development research and praxis can be obtained from the systematic application of the radical pragmatic philosophy of science proposed in this thesis. Similarly, it would be worthwhile to revisit the history of human-computer interaction research and praxis from the orthodox cognitivist and onwards. Also, it seems promising to follow the ambition of Card et al. (1983) to build an engineering psychology, not based on their mechanical materialism, but based on the dialectical materialism of activity theory; contributing to real applicability of activity theory in the field of human-computer interaction. Generally Wartofsky's dialectical materialism provides a basis for a stream of new insights. E.g. further application of the notion of secondary artefacts in relation to current theories in the field of computer supported cooperative work based on the notion of articulation work seems promising.

The level of basic theory presents intriguing issues for future work. The primary problem in applying activity theory in systems development research has been the dependency on historical determinism, through the linear and vertical concept of the zone of proximal development; but as indicated by Engeström (1996), activity theory is moving away from this dependency. However, it is not clear whether the installation of a horizontal concept of development, etc., will turn activity theory into a relativistic mess of theoretically inconsistent babel, or it will be possible to maintain activity theory as an engaged and value-based foundation. The question is if the rejection of the uni-directional concept of development implies the general rejection of any notion of progress. The answer is no. A long tradition for non-utopian Marxisms exist, e.g. the Frankfurter school (see Adorno (1985) for an account on the notion of progress). Such positions will be of great importance for activity theory in the years to come. Related to the issue of the directionality of development is the role of the individual. Wartofsky's concept of off-line loop indicates a way to go in understanding innovation and creativity in materialist terms without insisting on collectivity as the only valid kind of dynamic force (e.g. Kuutti 1989).

Aesthetics is cultural reflection freed from purpose, the societal experience; aesthetics is the spring of innovation and emancipation. Thus, studies of the creation of art, will be a possible source of insights into the very basics of creativity, innovation and emancipation. The obvious difficulty in such an endeavour is to "see" what goes on in the genius mind of the artist, therefore groups of artists, e.g. theatres using a wide spectrum of technologies and an experimental mode working could be a fruitful laboratory. In the same way, multimedia projects together with artist who are non-computer experts could yield insights into this issue. Within the Danish National Centre for IT-research (CIT), we are moving in this direction in current pilot projects.

In this thesis, design has been described as a zone where heterogeneous praxes meet for a while to do something meaningful together, and the arte-

facts mediating this meeting have been analysed. The vocabulary applied seems promising in the further study of such boundary zones; not only in design, but also in relation to collaboration and experience in virtual rooms and common information spaces.

References

- Adorno, T. W. (1985). Fremskridt [Progress]. In Schanz, H.-J. & Thomsen, H.-J. (eds.). *Th. W. Adorno og det æstetisk moderne*. Aarhus: Modtryk. pp. 25-43.
- Andersen, N. E., F. Kensing, J. Lundin, L. Mathiassen, A. Munk-Madsen, M. Rasbech & P. Sørgaard (1990). *Professional systems development*. New York: Prentice Hall.
- Andersen, P. B. (1990). *A Theory of Computer Semiotics*. Cambridge: Cambridge University Press.
- Argyris, C. & Schön, D.A. (1978). *Organizational Learning: A Theory of Action Perspective*. Reading, MA.: Addison-Wesley Publishing Company.
- Bannon, L. & Bødker, S. (1991). Beyond the Interface: Encountering Artifacts in Use. In Carroll, John M. (ed.). *Designing Interaction*. Cambridge: Cambridge University Press. pp. 227-253.
- Bannon, L. J. & Kuutti, K. (1993). Searching for Unity among Diversity: Exploring the "Interface" Concept. In Ashlund, Stacey et al. (eds.) *INTERCHI '93: Conference on Human Factors in Computing Systems INTERACT '93 and CHI '93 Bridges Between Worlds*. New York, N.Y.: ACM Press. pp. 263-268.
- Bansler, J. (1987). *Systemudvikling: teori og historie i skandinavisk perspektiv*. Lund: Studentlitteratur.
- Bansler, J. (1989). Systems Development Research in Scandinavia: Three theoretical schools, *Scandinavian Journal of Information Systems* vol. 1, Aug. 1989, pp. 3-20.
- Bardram, J. E. & O. W. Bertelsen (1995). Supporting the Development of Transparent Interaction. In Blumenthal, Gornostaev, & Unger (eds.). *Human-Computer Interaction. 5th. International Conference, EWHCI '95 Moscow, Russia, July 1995. Selected Papers*. Berlin: Springer Verlag (LNCS 1015). pp. 79-90.
- Bertelsen, O. W. (1993). *Når Uret Blir Ret: et litteraturstudium til belysning af muligheden for teori i human-computer interaction*. [A study of the literature aiming at a clarification of the possibilities of theory in human-computer interaction], Unpublished masters thesis, Aarhus University.
- Bertelsen, O. W. (1994a). An investigation into the design artefact concept as a vehicle for systems development research. In *Proceedings of the 17th*

Information systems Research seminar In Scandinavia IRIS 17, Syöte, Finland, August 1994. pp. 715-125.

Bertelsen, O. W. (1994b). Fitts' Law as a Design Artefact: A Paradigm Case of Theory in Software Design. In Blumenthal, Gornostaev, & Unger (eds.). *Human-Computer Interaction. 4th. International Conference, EWHCI '94 St. Petersburg, Russia, August 1994. Selected Papers.* Berlin: Springer Verlag. pp. 11-18.

Bertelsen, O. W. (1996a). Contradictions in the Festival Project - Activity systems, obstacles and dynamic forces in design. In Dahlbom, Ljungberg, Nuldén, Simon, Sørensen & Stage (eds.). *Proceedings of the 19th. Information systems Research seminar in Scandinavia, (IRIS 19), 10-13 August 1996, at Lökeberg, Sweden.* pp.597-612.

Bertelsen, O. W. (1996b). The Festival Checklist: design as the transformation of artefacts. In Blomberg, J., Kensing, F. & Dykstra-Erickson (eds.). *PDC '96, Proceedings of the Participatory Design Conference.* Palo Alto: Computer Professionals for Social Responsibility. pp. 93-101.

Bertelsen, O. W. (1996c). Organisational learning is crystallised into artefacts, position paper for workshop on organisational learning at CSCW'96. In *SIGOIS Bulletin* vol. 17, no. 3, December 1996. pp. 37-39.

Bertelsen, O. W. (1997a). Understanding objects in use-oriented design. In Braa, Kristin & Monteiro, Eric (eds.). *Proceedings of the 20th Information systems Research seminar in Scandinavia.* Oslo, 1997. pp. 311-324.

Bertelsen, O. W. (1997b). Toward a Unified Field of SE Research and Practice. In *IEEE Software*, vol. 14, no. 6, November-December 1997. pp. 87-88.

Biering-Sørensen, S., et al. (1988). *Håndbog i struktureret programudvikling.* Copenhagen: Teknisk Forlag.

Bisgaard, O., Mogensen, P., Nørby, M., Thomsen, M. (1989). *Systemudvikling som lærevirksomhed, konflikter som basis for organisational udvikling.* Aarhus: DAIMI IR-88.

Bjerkness, G. (1992). Dialectical Reflection in Information Systems Development. In *Scandinavian Journal of Information Systems*, vol. 4. pp. 55-79.

Blumenthal, B. (1995). Industrial Design and Activity Theory: A New Direction for Designing Computer-Based Artifacts. In Blumenthal, Gornostaev, & Unger (eds.). *Human-Computer Interaction. 5th. International Conference, EWHCI '95 Moscow, Russia, July 1995. Selected Papers.* Berlin: Springer Verlag (LNCS 1015). pp. 1-16.

Borum, F. & Enderud, H. (1981). *Konflikter i organisationer: - belyst ved studier af edb-systemarbejde.* København: Nyt Nordisk Forlag.

- Bratteteig, T. & Øgrim, L. (1994). Soft Dialectics—Structured Handling of Problem Situations in System Development, Baets (ed.). *Proceedings of the Second European Conference in Information Systems, Nijenrode University, Breukelen, April 28-29*. pp. 681-690.
- Brooks Jr., F. P. (1986). No Silver Bullet – Essence and Accident in Software Engineering. In H.-J. Kugler (ed.). *Proceedings of IFIP tenth World Computing Conference*. Amsterdam: Elsevier Science B. V. pp.1069-1076.
- Burrell, G. & Morgan, G. (1979). *Sociological Paradigms and Organisational analysis*. London.
- Button, G. & Sharrock, W. (1994). Occasioned practices in the work of software engineers. In Jirotko, M. & Goguen, J. (eds.). *Requirements Engineering Social and Technical Issues*. London : Academic Press. pp. 217-240.
- Bærentsen, K. (1989). Menneske og maskine [Man and Machine]. In Hedegaard, Hansen & Thyssen (eds.). *Et virksomt liv [An active life]*. Aarhus: Aarhus University Press. pp. 142-187.
- Bødker, S. (1991). *Through the Interface: a human activity approach to user interface design*. Hillsdale, N.J.
- Bødker, S. & Mogensen, P. (1993). One woman's job is another man's articulation work - an essay about the design of computer support for cooperative work. In Robinson, M. & Schmidt, K. (Eds.). *Developing CSCW Systems: Design Concepts. Report of the CoTECH WG4*. pp. 149-166.
- Bødker, S., P. Ehn, J. Kammergaard, M. Kyng, & Y. Sundblad (1987). A UTOPIAN Experience: On Design of Powerful Computer-Based Tools for Skilled Graphic Workers. In Bjerknes, G., P. Ehn, & M. Kyng (eds.). *Computers and Democracy*. Aldershot UK: Avebury. pp. 251-279.
- Bødker, S., Knudsen, J. L., Kyng, M., Ehn. P. , Madsen, K. H. (1988). Computer Support For Cooperative Design. In *Proceedings of CSCW '88*. pp. 377.394.
- Bødker, S. & Grønbæk, K. (1996). Users and designers in mutual activity: An analysis of cooperative activities in system design. In Engeström, Y. & Middleton, D. (eds.). *Cognition and Communication at Work*. Cambridge: Cambridge University Press. pp. 130-158.
- Card, S.K.; English, W.K. & Burr, B.J. (1978). Evaluation of mouse, rate-controlled isometric joystick, step keys, and text keys for text selection on a CRT. In *Ergonomics* vol. 21. pp. 601-613.
- Card, S. K.; Moran, T. P. & Newell, A. (1983). *The Psychology of Human-Computer Interaction*. Hillsdale NJ.

- Carlsen, J., H.-J. Schanz, L.-H. Schmidt & H.-J. Thomsen (1984). *Karl Marx og den moderne verden* [Karl Marx and the modern world]. Copenhagen: Gyldendal.
- Carnap, R. (1935). The rejection of Metaphysics. In White, M. (ed.). *The age of analysis*. New York 1955. (Excerpt of Philosophy and Logical Syntax, London 1935).
- Carroll, J. M., Kellogg, W., Rosson, M. B., (1991). The Task Artifact Cycle. In Carroll, J. M., (ed.). *Designing Interaction: Psychology at the Human-Computer Interface*. Cambridge: Cambridge University Press. pp. 74-102.
- Chapanis, A. (1967). The Relevance of Laboratory Studies to Practical Situations. In *Ergonomics*, 1967, vol. 10 no. 5, pp. 557-577.
- Checkland, P. (1981). *Systems Thinking, Systems Practice*. Chichester: John Wiley & Sons.
- Christiansen, E. (1988). *Den Realistiske Vision: et humanistisk-datalogisk perspektiv på systemudvikling* [The Realistic Vision. A Humanistic Computer Science approach to Systems Development], Unpublished ph.d.-thesis. Aalborg University: institut for kommunikation.
- Coad, P. & E. Yourdon (1990). *Object-Oriented Analysis*. Engelwood Cliffs N.Y.
- Ehn, P. & D. Sjögren (1991). From system description to script for action. In Greenbaum, J. & M. Kyng (eds.). *Design at work: cooperative design of computer systems*. Hillsdale: LEA. pp. 241-268.
- Ehn, P. & Kyng, M. (1987). The Collective Resource Approach to System Design. In Bjercknes, Ehn, & Kyng, *Computers and Democracy*. Aldershot: Avebury. pp. 17-58.
- Ehn, P. (1988). *Work-oriented Design of Computer Artifacts*. Falköping: Arbejdslivscentrum.
- Engelsted, N. (1989). *Pesonlighedens almene grundlag I*. Aarhus: Aarhus University Press.
- Engeström, Y. (1987). *Learning by expanding: an activity-theoretical approach to developmental research*. Helsinki: Orienta-Konsultit Oy.
- Engeström, Y. (1996). Development as breaking away and opening up: a challenge to Vygotsky and Piaget. In *Swiss Journal of Psychology*, vol. 55 no. 2-3, pp. 126-132.
- Feyerabend, P. (1975). *Against Method*. London.
- Fitts, P. M. (1954). The information capacity of the human motor system in controlling the amplitude of movement. In *Journal of Experimental Psychology* vol. 47, no. 6. pp. 381-391.

- Gediga, G. & Wolff, P. (1989). On the applicability of three basic laws to Human-Computer Interaction, *MBQ* 11/89. Osnabrück.
- Gilbreth, F. B. (1911). *Motion Study*. New York.
- Gillan, D.J.; Holden, K.; Adam, S.; Rudisill, M. & Magee, L. (1990) How does Fitts' Law Fit Pointing and Dragging?. In *Proceeding of CHI'90 Conference on Human Factors in Computing Systems*. New York: ACM. pp. 227-234.
- Greenbaum, J. & Kyng, M. (eds.). (1991). *Design at Work*. Hillsdale, N.J.: LEA.
- Grønbaek, K. & Knudsen, J.L. (1992). Tools and Techniques for Experimental System Development. In *Nordic Workshop on Programming Environments, Tampara, Finland, January, 1992*. (no pagination).
- Henderson, A. & Kyng, M. (1991). There is no place like home: continuing design in use. In Greenbaum, J. & Kyng, M. (eds.). (1991), *Design at Work*. Hillsdale, N.J.: LEA. pp. 219-240.
- Hirschheim, R. & Klein, H. (1989). Four paradigms of information systems development. In *Communications of the ACM*, vol. 32, no. 10. pp. 1199-1216.
- Holbæk-Hansen, E., Håndlykken, P. , & Nygaard, K. (1975). *System Description and the Delta Language*. Oslo: Norwegian Computing Center.
- Jones, J. C. (1988). Softecnica. In Thackara, J. (ed.). *Design after Modernism*. London. pp. 216-226.
- Jungk, R. & Müllert, N. (1987). *Future Workshops: How to create desirable futures*. London: Institute for Social Inventions.
- Kapor, M. (1991). A Software Design Manifesto: Time for a Change. In *Dr. Dobbs's Journal*, no. 172 , Jan. 1991, pp. 62-68.
- Knudsen, J. L. (1994). Teaching object-oriented methodology and languages. In Knudsen, J.-L-, Löfgren, M., Madsen, O.L. & Magnusson, B. (eds.). *Object-Oriented Environments: the Mjølner Approach*. New York: Prentice Hall. pp. 578-600.
- Kuhn, T. S. (1962). *The structure of scientific revolutions*. Chicago.
- Kuutti, K. (1989). Flaws in rationalist thinking, exact creativity and information systems. In Bødker, S. (ed.). *Proceedings of the 12th IRIS seminar, Skagen, Denmark, August 1989*. Aarhus: DAIMI PB- 296-II. pp. 373-392.
- Kuutti, K. (1991). Activity theory and Its Applications to Information Systems Research and Development. In Nissen, H.-E., Klein, H. K., & Hirschheim, R. (eds.). *Information Systems Research: Contemporary Approaches & Emergent Traditions*. Amsterdam: North-Holland. pp. 529-549.

- Kyng, M. (1991). The System Work Group, Computer Science Department, Aarhus University. In S. P. Robertson, G. M. Olson, & J. S. Olson (Eds.). *Human Factors in Computing Systems, Reaching Through Technology*. New Orleans, Louisiana, ACM Press. pp. 477-478.
- Landauer, T. K. (1991). Let's get real: a position paper on the role of cognitive psychology in the design of humanly useful and usable systems. In Carroll, J.M. (ed.). *Designing Interaction: Psychology at the Human-Computer Interface*. Cambridge: Cambridge University Press. pp. 60-73.
- Laursen, B; L.L. Jensen; L.O.W. Bertelsen; T.A.N. Troelsen (1990). *Dokumentation & Kommunikation i oliebudgetprojektet*, [Documentation and Communication in the oil-budgeting project.]. unpublished case study report, Computer Science Department at Aarhus University.
- Leontjev, A. N. (1978). *Activity, consciousness, and personality*. Englewood Cliffs NJ: Prentice Hall.
- Leontjev, A. N. (1981). *Problems of the development of the mind*. Moscow: Progress.
- Madsen, K. H. (1996). Initiative in Participatory Design. In Blomberg, J., Kensing, F. & Dykstra-Erickson (eds.). *PDC '96, Proceedings of the Participatory Design Conference*. Palo Alto: Computer Professionals for Social Responsibility. pp. 223-230.
- Madsen, O. L., Møller-Pedersen, B. & Nygaard, K. (1993). *Object-Oriented Programming in the BETA Programming Language*. Wokingham, England: ACM Press/Addison-Wesley Publishing Company.
- Mammen, J. (1989). The relation between subject and object from the perspective of activity theory. In Engelsted et al. (ed.). *Essays in General Psychology*. Aarhus: Aarhus University Press. pp. 71-94.
- Mao Tsetung (1937). Om Modsigelse [On contradiction]. In Mao Tsetung (1978). *Udvalgte Værker af Mao Tsetung*, bind 1. Copenhagen: Forlaget Oktober. pp. 261-292.
- Marx, K. (1845). Theses on Feuerbach, my translation from Danish version. In Johs. Witt-Hansen (ed.). (1974). *Karl Marx Skrifter i udvalg, Den tyske ideologi, Filosofiens elendighed*. København: Rhodos.
- Marx, K. (1962). *Das Kapital*, vol. I. Berlin: Dietz Verlag .
- Marx, K. & Engels, F. (1848). *Det kommunistiske manifest* [The communist manifesto]. København: Rhodos (unknown publication year).
- Mathiassen, L. (1981). *Systemudvikling og Systemudviklingsmetode* [in Danish: System development and system development method]. Aarhus: Computer Science Department DAIMI PB-136 .

- Meister, D. & Farr, D. E. (1967). The Utilization of Human Factors Information by Designers. In *Human Factors*, vol. 9. pp. 71-87.
- Mogensen, P. (1992). Towards a Prototyping Approach in Systems Development. In *Scandinavian Journal of Information Systems*, vol. 4. pp. 31-55.
- Mogensen, P. (1994). *Challenging Practice: an Approach to Cooperative Analysis*, Ph.D. thesis, Aarhus University: DAIMI PB-465.
- Mørch, A. (1997). Three Levels of End-User Tailoring: Customization integration, and Extension. In M. Kyng & L. Mathiassen (eds.). *Computers and Design in Context*. Cambridge, MA.: MIT Press. pp. 51-76.
- Nardi, B. (ed.). (1995). *Context and Consciousness: Activity Theory and Human Computer Interaction*. Cambridge MA.: MIT Press.
- Naur, P. (1985). Programming as theory building. In *Microprocessing and Microprogramming* vol. 15, pp. 253-261.
- Newell, A. & Card, S. K. (1985). The Prospects for Psychological Science in Human-Computer Interaction. In *Human Computer Interaction 1985* vol.1, pp. 209-242.
- Norman D. A. (1991). Cognitive artifacts. In Carroll, J.M., (ed.). *Designing Interaction: Psychology at the Human-Computer Interface*. Cambridge: Cambridge University Press. pp. 17-38.
- Norman, D.A. & Draper, S. (eds.) (1986). *User-centered system design: New perspectives on human-computer interaction*. Hillsdale, N.J.: LEA.
- Popper, K. (1973). Videnskaben: gisninger og gendrivelsler [conjectures and refutations]. In *Kritisk rationalisme*. Copenhagen.
- Raeithel, A. (1991). Activity Theory as a Foundation for Design. In Floyd, Züllighoven, Budde & Keil-Slavik (eds.). *Software development and reality construction*. Berlin: Springer Verlag. pp. 391-415.
- Schanz H-J. (1994). *Karl Marx i tilbageblik efter murens fald* [Karl Marx in retrospect after the collapse of the wall]. Aarhus: Modtryk.
- Shannon, C. E. & Weaver, W. (1949). *The mathematical theory of communication*. Illinois.
- Sørgaard, P. (1988). Object-Oriented Programming and Computerised Shared Material. In *Proceedings of ECOOP, Oslo Norway, August 1988*. Berlin: Springer Verlag. pp 319-334.
- Star, S. L. (1989). The Structure of Ill-Structured Solutions: Boundary Objects and Heterogeneous Distributed Problem Solving. In Gasser, Les & Michael N. Huhns *Distributed Artificial intelligence, volume II*. London: Pitman Publishers, pp. 37-54.

- Strauss, A., Fagerhaug, S., Suczec, B., & Wiener, C. (1985). *Social organization of medical work*. Chicago: The University of Chicago Press.
- Taylor, F. W. (1916). *Scientific management*. New York.
- Trigg, R., Moran, T.P. , & Halasz, F.G. (1987). Adaptability and Tailorability in NoteCards. In Bulinger & Shackle (eds.). *Human-Computer Interaction - INTERACT '87*. Amsterdam: Elsvier/North-Holland. pp. 723-728.
- Vygotsky, L. (1978). *Mind in society: The development of higher mental processes*. Cambridge, MA: Harvard University Press.
- Wartofsky, M. W. (1973). Perception, representation, and the forms of action: toward an historical epistemology. In Wartofsky, M. W., *Models*. Dordrecht: D. Reidel Publishing Company, 1979. pp. 188-210.
- Wartofsky, M. W. (1977). *Feuerbach*. Cambridge: Cambridge University Press.
- Winograd, T. & Flores, F. (1986). *Understanding Computers and Cognition -- a new foundation for design*. Norwood, N.J: Ablex.
- Yourdon, E. (1982). *Managing the system life cycle*. New York: Yourdon Press.

PART II

First paper

An investigation into the design artefact concept as a vehicle for systems development research

Abstract: System development is studied from the viewpoint of artefacts utilised during the design process. Different aspects of design artefacts are discussed based on a pragmatic classification of design activities and a philosophical notion of perception and representational artefacts. Statements about the relation between theory and design are made based on materialist pragmatism.

Introduction

In the last years the development of tools and environments for experimental system development has been the main issue for the system development research group at Aarhus University. This can be seen as a natural progression of the critical Scandinavian tradition: In the early projects (NJMF, DEMOS, DUE) the focus was on changed motives for intervention in system design. The later projects (UTOPIA, MARS) realised that the new motives induced a demand for new ways of acting (methods, techniques) in design. Now we (the Devise project) understand that the design life we want to live requires a new house — new tools and environments.

In this paper I look at system development through the artefacts utilised in the design process. I will call such artefacts *design artefacts*, Admittedly this concept is somewhat outlandish and very wide in definition. The reason for talking about design artefacts instead of just design tools or system development environments, is twofold. Firstly it emphasises that system development is a human activity like many other activities, and thus mediated by certain artefacts. Secondly the concept indicates that design is different

from use, and that the activity of deliberate transformation of the environment for our daily life contains elements which are not to the same extent present in our day to day praxis. In this paper design is opposed to use, and should not be confused with “design phase” (as opposed to analysis, and implementation) defined in many system development methods.

Design

Design is an activity where a designing subject shapes the design object by means of some design artefacts. Artefact does not only denote “man made things” as the dictionary suggests, but further points at things as mediators of human activity. The artefact concept thus emphasises that things are what they are due to their use rather than their physical features. In general artefacts are mediating either a relation between a subject and an object, or a relation between subjects. An artefact mediates an activity when the focus of the acting subject is on the object/subject for the activity rather than on the artefact. If focus is on the artefact, this become the object of the activity, and other artefacts come to mediate the activity. (For further explanation see e.g. Leontjev 1979).

The design object is the artefacts that the design process changes or creates. The design object is parts of the deliberately shaped environment, or conditions for human life: houses, cars, word-processors, and furniture, but not food, movies, news papers and toilet paper. The designing subject is almost always a collective subject. Some of the members of this subject can be professional designers who designs the conditions for the lives of somebody else, and others can be members of the praxis that the design object is intended to mediate. Design artefacts are artefacts that mediate the design activity, they are utilised but not consumed during the process. They serve as conditions or environment for the design process. Design artefacts can thus be opposed to materials.

In the perspective of design artefacts, system development consists of three elements, or main design functions mediated by the design artefacts. The three elements are: (1) getting knowledge and understanding about what is designed, (2) communicating during this process, and (3) shaping the considered world. Examples of design artefacts are programming languages, CASE-tools, specification standards, and systems development methods. Often we think of design artefacts as supporting just one of the design elements: an editor is used for writing code, a future workshop is used in order to understand the problem domain, memos are used for communication, and so on.

The computer is a symbol manipulating device, and thus design of computer artefacts can be seen as construction of signs. Most human-computer interaction research has tried to find the mechanics of these signs, in order to optimise the collaboration between man and machine. Computer semiotics (Andersen 1990) has, however, pointed out that software and hardware are not signs but only potential carriers of signs, or *generators of signs*, and that the signs are constituted by the use praxis. This implies that as designers we don't know what we are constructing, and thus the fundamental lack of experience that is a basic condition for system development (Kyng 1994) becomes twofold: we don't know the problem domain, and it is impossible to know what we are building before the design process has ended.

Three elements of design

As stated above system development consists of the three elements: *conception*, *communication* and *construction*.

System development is a process aiming at the *construction* of software and the environment for its use. In this way we are talking about system development as a programming process. The artefacts mediating the relation between designers and the design object seen as technical construction, are programming environments, CASE tools etc. The artefacts mediating the constructive aspects of design also mediate communication and perception. The use of a CASE tool makes the designers conceive the design object with the concepts supported by the tool, and will thus guide the process in certain directions. When looking at design as construction, the benefits of object-oriented programming languages become the possibilities of code reuse, the high degree of maintainability, and the stability of "code structure".

Design is a co-operative enterprise, where different people with different professional backgrounds and different motives are engaged in creating something new. This is the reason why communication is essential in system development. The design artefacts mediate system development as *communication*, both as explicit means of communication (e.g. status reports and specifications) and as means for the sharing of experiences, insights, and visions about the design object. A prototype mediates communication when users through their exploration of the prototype, yield knowledge about the use context to designers, and when designers express their new insights by way of continuously changing the prototype.

Design is *conception* of the present and of the new. Users and designers achieve new understanding of the existing world and rooted in this they conceive a new world that transcend the old. Design artefacts mediate learning and conception. A functional programming language for example facilitates

the understanding of the design object as transformation of data streams. A prototype induces new visions and knowledge when it confronts the possible and the existing. In a positivist epistemology, the conceptional elements of design will be the passive operations of mere inspection and evaluation. In such a view the above analysis of the three design elements would not say anything about design, and we could say with Card et al. (1980) that “Scientific models don't eliminate the design problem, but only help the designer control the different aspects.”. In contrast to this pessimism I claim that it *is* possible to base design on scientific studies (although not ideal natural scientific) of the design object and of design.

It is the conceptional and communicative elements that separate system development from programming, but as Naur (1985) has pointed out there is more to programming than writing machine executable code. Naur's thesis is that a theory about what the program does, and how it does it, is built simultaneously with the production of the executable code. The theory can't be written down or otherwise formalised, it is only accessible to the programmers. In this notion, theory becomes a mysterious, immaterial construct. Naur's conclusions offer no understanding of how knowledge can be transferred between different groups of people, or what (historical) human praxis is. Naur fails to see that everything man does is rooted in his societal context, and that the technical process of constructing a program is subsumed under its evolving social praxis. Sørgaard (1988) adopts the same mistake in his discussion of the relation between programming environments and system development environments, by claiming that all co-operative work including systems development, is an aggregate of individual work (op sit p.14). This is the idea that human life can be understood by first considering human biology, then the mind of the individual, and last the society. But the human mind can't be understood as a biological phenomenon, and sociality is not an aggregate of individuals. I will not argue for historical determinism or against the existence of some kind of free will. But I will claim that those aspects of human behaviour which are related to system development can only be understood by looking at man as a product of a socio-historical process.

If the classification of system development into the three elements conception, communication and construction is compared to the MARS-framework (Andersen et al. 1990), the first thing to be noticed is that my scheme does not explicitly deal with the so-called “process oriented” components i.e. planning, evaluation, and regulation. Although these “process oriented” components are located in the communication category, it is obvious that project management is one of the major bottlenecks in practical system development, and that it should be paid careful attention to by practitioners. But in trying to understand the basic features of design I think it is unnecessary to deal with these aspects explicitly.

Representational artefacts

While it is fairly simple to see how the communicational and constructional elements of design are mediated by various artefacts, it is far more complicated to see and appraise how the conceptional element is mediated by these artefacts. In order to clarify that question the notion of secondary artefacts is adopted from an essay on perception by Marx Wartofsky (1973, p.202):

...what constitutes a distinctively human form of action is the creation and use of artifacts, as tools, in the production of the means of existence and in the reproduction of the species. *Primary* artifacts are those directly used in this production; *secondary* artifacts are those used in the preservation and transmission of the acquired skills or modes of action or praxis by which this production is carried out. Secondary artifacts are therefore *representations* of such modes of action, and in this sense are *mimetic*, not simply of the *objects* of an environment which are of interest or use in this production, but of those objects as they are acted upon, or of the modes of operation or action involving such objects.

Design artefacts are primary artefacts as well as secondary artefacts. As primary artefacts they mediate construction and communication, and they also mediate the creation of various artefacts like charts and diagrams which belong to the conceptional element. When seen as primary artefacts, design artefacts can be analysed with the same concepts that we use in the analysis of artefacts in general. From this point of view CASE-tools and debuggers are very similar to cars and frying-pans. Bødker's (1991) activity theory based analysis of the human-computer interface, is an analysis of computer artefacts as primary artefacts i.e. how the computer artefacts mediate either relations to objects (instrumental) or relations to other subjects (communicational).

It is as secondary artefacts, or rather as generators of secondary artefacts the design artefacts are constituted. Ehn and Sjögren's (1991) notion of design artefacts is in some sense related to the concept of secondary artefacts. They use the concept to denote material representations produced during the design process, not as part of the final product, but as vehicles for the generation of (visions of) the new. Such artefacts, belonging to a specific project, could be classified as local secondary artefacts, in order to separate them from "global" secondary artefacts like theories, methods, and the like, and to separate them from general design artefacts.

If we adopt the notion of secondary artefacts, the models and descriptions made in system development appear to be conventional reminders rather than one-to-one reflections of the world. Thus modelling artefacts should not

be evaluated with “naturalist” criteria, but rather by looking at the actual design praxis made possible by the formalism.

Naur’s idea of programming as theory building can be reformulated with the notion of secondary artefacts. In this perspective the program is an externally embodied representation of the modes of action upon the world that the programmers have discovered during the programming process. Naur is right in stating that there is more to programming than the production of an “objective” specification, but this extra thing is not impossible to transfer. In contrast such external embodiments are basically vehicles for the process of making modes of action, generally available for other members of the human species. Naur describes a case where a team of programmers successfully transferred the “theory” of a program to another team, which they made a revision of the program together with. Naur explains this by claiming that the new team experienced the creation of the program, but that was not the case since the program was not totally re-constructed. If the “theory” is understood as a secondary artefact we can avoid this mystification, and instead see the theory as a “material” shareable part of social praxis.

Primary artefacts as generators of secondary artefacts

Though there is more to system development than mere programming, system development may be seen as a process aiming at the production of programs that can be run on a computer. Thus the primary artefacts in system development are those things normally referred to as programming environments. The tool par excellence, the hammer, is primarily an instrument for construction e.g. driving nails and splitting bricks. But at the same time the hammer is used as an instrument for our perception of the world. When the carpenter drives the nail he also learns something about the specific piece of wood. Norman’s (1991) concept of *cognitive artifacts* does not offer a suitable explanation of this aspect of the use of tools. Norman builds his notion of cognitive artefacts on a division between evaluation and execution. First we test on the world, then we operate, then we test to see if we can exit the loop. This division has likewise been the core of the work science tradition founded by Gilbreth (1911) and Taylor (1916) — the stopwatch man evaluates operations and the worker performs operations. Understanding and changing the world can only be seen as separate in degenerated contexts like the psychological laboratories or the factory assembly lines. The real world is moving and unpredictable and we have to dig the holes for the foundation pillars in the mud before we can know where to build our house.

Programming tools serve as mediators for the creation of the running system. But at the same time the tools mediate the perception and understanding of the existing and the future worlds. In this way simple tools like hammers can be seen as generators of secondary artefacts. These secondary artefacts are *not pictures* of the hammer or the wood, but representations of modes of acting with/upon hammer and wood.

The applicability of elements from structured analysis (Yourdon 1982) in communication and conception, together with users depends highly on the designers abilities to transform e.g. a context diagram produced with a CASE-tool, into a secondary artefact. Laursen et al. (1990) report on two different situations in the same organisation. The two situations comprised almost the same persons but in different development projects. In the first situation a context diagram (called the “the sunflower” by the project group) was successfully applied as a common point of reference shared by all members of the project group. In the second situation the designers tried to apply ER-diagram output from a CASE-tool in discussing aspects of the application area with the users. In this situation the diagrams failed as generators of shared secondary artefacts. This suggests that different types of diagramming techniques (as primary artefacts) are differently suited as generators of secondary artefacts. But what are these differences based on? What makes a primary artefact a good generator of secondary artefacts?

The schools of “ontological” object-orienting (e.g. Coad and Yourdon 1990) suggest that the advantage of the object-oriented methods is that the world *is* object-oriented. This idea is based on a world view that fails for the parts of the world that contain human beings, because laws of mechanics and common sense logic only are applied by human beings when they are engaged in detached reflection (Ehn 1988, Suchman 1986). But how can we then defend the widely accepted, although never empirically supported, idea that object-oriented methods are much better than other known methods for programming and software specification?

Object-oriented methods have much in common with LEGO bricks. The good thing about LEGO bricks is not that the world is LEGO-oriented or that human beings think about the world, and have visions in LEGO brick-like concepts. The good thing about LEGO-bricks is that they are easy to construct with and that it is possible to re-combine existing constructions. Using LEGO bricks it is possible to develop the vision of a new space vehicle shaped in a way that totally transcends the limitations of LEGO. The suitability of LEGO bricks for the construction of space vehicles is not due to the LEGO-like form of the world of space vehicles, but a result of LEGO-bricks’ potentials as *generator of secondary artefacts*. I think that the virtues of object-oriented methods in system development are comparable to those of LEGO in the field of flying saucer design. Thus I will agree with Morten Kyng (1994) in saying: “By means of object-orientation it is possible to design

computer artefacts such that it is easier for users to create relations between these artefacts and that which the artefacts is about”. And I will furthermore suggest that this, and not some kind of naturalism, may be the background for the virtues of *object-oriented design artefacts*.

Science in design

Scientific theories can be understood as secondary artefacts pretending to transcend the limitations of their historical context. Cognitive science based, orthodox human-computer interaction research, believes that design tools can be built by making approximate versions of the general theories of human cognition (Card et al. 1983). Although the rigid hard science vision of Newell and Card (1985) is not shared by everybody in the human-computer interaction community, the ideals of the natural sciences are widely accepted. Universal disinterested knowledge is seen as the only alternative to unsystematic design by rules of thumb and ad hoc procedures. This general lack of theories that are pragmatically rooted in praxis has been destructive for the entire field of human-computer interaction (Bertelsen 1993).

In general scientific theories are secondary artefacts, but when specific tools for calculation are extracted from a theory, this extract becomes a primary artefact. Fitts’ law is probably the most cited result from experimental psychology in the field of human-computer interaction, and is in addition very suitable as a paradigm case for the study of the relation between theory and design (Bertelsen in press). Fitts’ law (Fitts 1954) states that there is a certain relation between how fast the human hand can be moved and the length and precision of the movement, and further that this relation is congruent with an ideal mathematical communication channel (Shannon 1994). The classical “Fitts’ law study” in the field of human-computer interaction was performed by Card, English, and Burr (1978) who compared the performance of various pointing devices in a text selection task. Later on the law was canonised as part of “the model human processor” by Card, Moran and Newell (1983).

As a design artefact Fitts’ law has played at least three different roles. As ontology, as tool for calculation, and as source of inspiration. In the human-computer interaction psychology by Card et al. (1983) Fitts’ law was a part of the *world view*. To play this role, Fitts’ law must be placed in a context where human beings are seen as mechanical devices, that is cognitive science. Landauer (1991) reports on the use of Fitts’ law as a *tool* for specific optimisations in the design process, which not necessarily implies cognitive science as a main perspective on HCI. Gillan et al. (1990) have used Fitts’ law as a *metaphor* for research on specific aspects of acting with a mouse. In

design this is equal to the use of the Fitts' effect as a thinking tool (Gediga and Wolff 1989). The borders between these three roles are not clear-cut. By thinking about interface problems in Fitts' law terms (the metaphor role), other views are excluded and Fitts' law draws the main perspective towards mechanistic reductionism. The use of a specific performance calculation serves as a thinking tool too. As a tool for calculation Fitts' law is a primary artefact, just as a microscope. Of course, the things we are able to see by using Fitts' law are irrelevant for design unless the calculations can be related to the use situation. As a metaphor, Fitts' law is a "global" secondary design artefact. As ontology, Fitts' law must be tied to the idea that the world is out there no matter what we do about it, *and* that it is possible to get universal knowledge about this world. In the later role Fitts' law and the entire cognitive science framework is a secondary artefact. To appreciate its validity, we have to ask if it helps us to shape the world in a desirable way. Studies of applications of Fitts' law in human-computer interaction show that universal knowledge abstracted beyond a specific context, can't be regarded as confident (Bertelsen 1994b).

What has surprised me is that similar conditions can be identified in the field of algorithm construction. In recent lectures at the department of computer science in Århus, Neil Jones and Jens Clausen (Department of Computer Science at the University of Copenhagen) have pointed out that theories of asymptotic complexity are insufficient as instruments for the identification and refinement of effective algorithms in practical situations. Clausen pointed out that evaluation of the different algorithms in a specific context is necessary. He further suggested a checklist for qualified choice of algorithms. The importance of these observations for this paper is that even in the very "exact" world of algorithm construction, theories can't be treated as "one-to-one" reflections of the problem domain e.g. the performance of a specific algorithm. General theories should rather be seen as artefacts facilitating deliberate intervention based on specific conceptions. Theories will be better understood if we think of them as secondary rather than primary artefacts.

Conclusion

In this paper I have introduced three analytical categories which can be applied in order to understand what is going on in system development, and I have further more discussed how the notion of secondary artefacts can yield an understanding of design artefacts. The discussion of the benefits of object oriented methods showed how this notion can provide us with new insights that transcended the naturalist tendency in object oriented modelling.

On the level of design theory the notion of secondary artefacts gives rise to a pragmatic materialism that can prevent the design sciences from driving into the ditches of relentless logical positivism, indifferent relativism, or foggy idealism. A practical approach to the construction of design artefacts based on the theoretical constructs presented above has not been discussed so far. One of the things that such a discussion may reveal is the need for a more explicit treatment of the “process oriented” component from the MARS-framework, in order to deal with issues like version control and resource allocation.

The focus on representations and modelling in contemporary system development literature, enforces a discussion of how human beings relate to their surroundings. We have to understand how this relation is different from the ways animals and machines relate to their surroundings. This paper hopefully contributes to that discussion.

Postscriptum

In the above mentioned essay by Wartofsky a third category of *tertiary artefacts* is also suggested. These artefacts relate to the ordinary worldly productive praxis in the same obscure way as dreams relate to the awake life. Tertiary artefacts have not been discussed in this paper, because it is difficult to relate this concept to design. Nevertheless, I am sure that aesthetics should be brought into this field. Not as rules about screen colouring or as application of Aristotelian poetics (Laurel 1986), but as a field of emancipatory, authentic praxis where unexpected things can happen. The basic aesthetic problem in design of computer artefacts, is that everything in an interface is planned, or should be. If something unexpected shows up on your screen then it is the result of bad design and it will cause immediate frustration. While the aesthetics of the modern world is constituted by a contingent stream of experiences (truck horns, TV-antennas, paint peeling of a wall), the world of computers only supplies us with over planned images. What is needed in this functional concrete dessert is more TV-antennas.

Acknowledgements

Thanks to Anders Mørch for discussions and for several corrections of my poor English, to Kim Halskov Madsen for demonstrating where I did not manage to get the message across, and to Mette Robak for late night proof reading.

References

- Andersen, N. E., F. Kensing, J. Lundin, L. Mathiassen, A. Munk-Madsen, M. Rasbech and P. Sørgaard (1990), *Professional systems development*, New York: Prentice Hall.
- Andersen, Peter Bøgh (1990), *A Theory of Computer Semiotics*, Cambridge: Cambridge University Press.
- Bertelsen, Olav W. (1993) *Når Uret Blir Ret... Et litteraturstudium til belysning af muligheden for teori i human-computer Interaction*. [A literature study aiming at a clarification of the possibilities of theory in HCI, in Danish], Unpublished masters thesis, Aarhus University.
- Bertelsen, Olav W. (1994) Fitts' law as a design artefacts, in *Proceedings of the 1994 East-West International Conference on Human-Computer Interaction*, Berlin: Springer Verlag.
- Bødker, Susanne (1991), *Through the interface: a human activity approach to user interface design*, Hillsdale, N.J: LEA.
- Card, S.K.; English, W.K. and Burr, B.J. (1978), Evaluation of mouse, rate-controlled isometric joystick, step keys, and text keys for text selection on a CRT, in *Ergonomics* vol.21, pp.601-613.
- Card, Stuart K., Thomas P. Moran and Allen Newell (1980), The keystroke-level model for user performance time with interactive systems, in *Communications of the ACM #7* vol.1980.
- Card, Stuart K., Thomas P. Moran and Allen Newell (1983) *The Psychology of Human-Computer Interaction*, Hillsdale NJ.
- Coad, Peter and Edvard Yourdon (1990), *Object-Oriented Analysis*, Engelwood Cliffs N.Y.
- Ehn, Pelle (1988), *Work-oriented Design of Computer Artifacts*, Falköping: Arbejdslivscentrum.
- Ehn, Pelle and Dan Sjögren (1991), From system description to script for action, in Greenbaum, Joan and Morten Kyng (eds.), *Design at work: cooperative design of computer systems..* Hillsdale: LEA.
- Fitts, Paul M. (1954), The information capacity of the human motor system in controlling the amplitude of movement in *Journal of Experimental Psychology* vol. 47, no. 6.
- Gediga, Günter and Wolff, Peter (1989), On the applicability of three basic laws to Human-Computer Interaction, *MBQ* 11/89, Osnabrück.
- Gilbreth, Frank B. (1911), *Motion Study*, New York.

- Gillan, D.J.; Holden, K.; Adam, S.; Rudisill, M. and Magee, L. (1990), How does Fitts' Law Fit Pointing and Dragging?, in *Proceeding of CHI'90 Conference on Human Factors in Computing Systems* (pp.227-234), New York.
- Kyng, Morten (1994), Making Representations Work, in L. Suchman (Ed.), *Representations of Work, HICSS-27*, p. 19-35, Hawaii.
- Landauer, Thomas K. (1991), Let's get real: a position paper on the role of cognitive psychology in the design of humanly useful and usable systems, in Carroll, J.M. (ed.), *Designing Interaction: Psychology at the Human-Computer Interface*, Cambridge.
- Laurel, Brenda K. (1986), Interface as mimesis, in Norman and Draper (eds.), *User centered system design*, Hillsdale, NJ: LEA
- Laursen, B; L.L. Jensen; L.O.W. Bertelsen; T.A.N. Troelsen (1990), *Dokumentation & Kommunikation i oliebudgetprojektet*, (Documentation and Communication in the oil-budgeting project, in Danish), unpublished case study report, Computer Science Department at Aarhus University.
- Leontiev, A.N. (1979), *Tätigkeit, Bewußtsein, Persönlichkeit*, Berlin (DDR).
- Naur, Peter (1985), Programming as theory building, in *Microprocessing and Microprogramming* vol. 15, pp. 253-261.
- Newell, Allen and Card, Stuart K. (1985) The Prospects for Psychological Science in Human-Computer Interaction, in *Human Computer Interaction 1985* vol.1, pp.209-242.
- Norman Donald A. (1991), Cognitive artifacts, in Carroll, J.M.,(ed.), *Designing Interaction: Psychology at the Human-Computer Interface*, Cambridge
- Shannon, Claude E. (1949), *The mathematical theory of communication*, Illinois.
- Suchman, Lucy A (1986), *Plans and situated actions*, Cambridge.
- Staunstrup, Jørgen (1979), *Besvarelse af obligatoriske opgaver*, Aarhus.
- Sørgaard, Pål (1988), *Programming Environments and System Development Environments*, Århus: DAIMI PB-252
- Taylor, Frederick Winslow (1916), *Scientific management*, New York.
- Wartofsky, Marx W (1973), Perception, representation, and the forms of action: toward an historical epistemology, in Wartofsky, M. W., *Models*, Dordrecht: D. Reidel Publishing Company, 1979.
- Yourdon, Edvard (1982), *Managing the system life cycle*, N.Y.: Yourdon Press.

Second paper

Fitts' Law as a Design Artefact: A Paradigm Case of Theory in Software Design

Abstract: Fitts' law is described and discussed as an example of use of theory in human-computer interaction design. The dichotomy between academic theory and applied theory is rejected and replaced by a radical pragmatic notion of theories as design artefacts. Different roles of theory in design are discussed.

Introduction

In the early years of interactive computing, designers relied solely on their intuition, some rules of thumb and a few guidelines. As the users of interactive systems turned from programmers to non-computer professionals, this situation became a problem —the systems were too difficult to use. Some saw in this an urgent need for a scientific foundation for HCI-design. The classical contribution to this vision is formed by the works by Card, Moran and Newell [3, 4], who believed that the future science of HCI should be based on cognitive psychology. They saw their task as one of making the bulk of academic experimental results applicable for designers. The vision was that this science had to be guided by the requirements set by the interface engineers, i.e. the theory should be operational rather than true in a more academic sense. The basic components in this applied science of the human-computer interface were task analysis, calculation and approximation. The idea was that the performance of a future human-computer system could be calculated from an analysis of the job which the system was intended to do. The special need for approximation in this field, compared to e.g. electrical engineering, was that the human system component was too

complex. The implicit epistemology underlying Card, Moran and Newell's vision is formulated with great clarity in Newell's last book:

Theories are approximate. Of course, we all know that technically they are approximate; the world can't be known with absolute certainty. But I mean more than that. Theories are also deliberately approximate. Usefulness is often traded against truth. Theories that are known to be wrong continue to be used, because they are the best available. Fitts' law is like that. How a theory is wrong is carried along as part of the theory itself. Grossly approximate theories are continuous launching pads for better attempts. Fitts' law is like that too. The scientist does the best he can with what he's got - and the engineer, who has to have answers, even more so. [13, p.14]

Newell doesn't reject the existence of a universal truth about human cognition, neither does he claim in principle that it is impossible to know this truth. The world *is* out there, but we construct approximate theories because it is too cumbersome to build an exhaustive theory, and because in many practical situations we are better off with operational rather than true theories. We can't reach the truth so we have to live with the useful. Newell's pragmatism is based on a dichotomy between applied theory (that is wrong but useful) and the truth. This position could be called methodological pragmatism.

In the field of HCI, the most prominent result from cognitive psychology is Fitts' law [6]. In this paper, I will use Fitts' law as a vehicle for a discussion of the role of theory in human-computer interaction, viewing it as a design artefact. Design artefacts are employed in the design process in order to support (or mediate) one or more of three design functions: gaining knowledge and understanding about what is to be designed, communicating during the process, and affecting the world. Examples of design artefacts are programming languages, CASE-tools, specification standards, systems development methods, and the like. Theories are thus constructed to help us master the world we are living in. Here I look at Fitts' law because it plays various interesting roles in the game of science and design.

Fitts' Law.

The goal of Fitts' work [6] was to make sense of experimental results about human motor performance that seemed to be mutually contradictory.

The need for a unifying concept of motor capacity is indicated by the apparent difficulty of reconciling many of the facts reported in the literature on motor skill. [6, p.382]

Thus Fitts' motivation was the purely academic one of making sense of some phenomena. Fitts' idea was that this could be obtained by realising that the capacity of the motor system (the relation between the movement-time, distance, and required precision) could be compared to the capacity of a communication channel [15], that is, a model where a sender codes signals onto a channel with a given bandwidth possibly perturbed by noise, after which a receiver gets the signal from the channel and decodes it. Fitts did not claim that mechanisms for coding etc. could be found in the motor system, but only that the limitations on performance had the same mathematical structure.

Fitts conceived the motor system as consisting of bones, muscles and nerves in the arm, as well as control mechanisms such as visual feedback. That is, the motor system consists of everything from “the signal in the brain” that initiates the movement to the resulting movement outside the subject. Fitts points out that one can only observe the total system of: receptor → neural channel (cognitive system) → effect, from the eye via the brain to the finger; you stimulates the eye and observes the resulting response in the hand. He works around this problem by applying a trick. By assuming that the neural channel consists of specific sub-channels in a chain, he sets up an experiment where everything except the motor system is eliminated. By using repetitive over-learned movements at the highest possible speed, he eliminates perception and cognitive processing from the chain, leaving the isolated capacity of the motor system (including the subjects' monitoring of the movement) to be measured. This was done in the so-called reciprocal tapping set-up.

Fitts defines the capacity of the human motor system as the ability to perform certain classes of movements in a uniform way. The time required to do a specific movement is inversely proportional to the capacity of the “motor channel”. This capacity can be described by an analogy to Shannon's:

Theorem 17: The capacity of a channel of band[width] W perturbed by white thermal noise of power N when the average transmitter power is limited to P is given by

$$C = W \log \frac{P + N}{N}$$

[15, p.100]

The obvious problem of this analogy is that it is impossible to get any information about the information capacity of the motor channel. Fitts does not see the information capacity of the motor system as a property of the motor system per se, but as a property of motor performance under given conditions. Thus the basis for Fitts' analysis is the amount of information required to accomplish a given movement. This quantity he denotes as the index of difficulty I_d , described by:

$$I_d = -\log_2 \frac{W_s}{2A} \text{ bits/response}$$

where W_s is the variability of the movement, and A is the amplitude (distance). Division by the time t it takes to do the movement yields a quantity that is constant for specific classes of movements. This is called the index of performance, $I_p = I_d/t$. Consequently, Fitts' law has the following form:

$$I_p = -\frac{1}{t} \log_2 \frac{W_s}{2A} \text{ bits/sec}$$

where I_p corresponds to Shannon's C , $1/t$ to the bandwidth of the motor system, and $W_s/2A$ to the signal to noise ratio, $(P+N)/N$ in Shannon's formula. The interesting point is that this equation is able to describe the empirical data. From this expression of I_p , it is simple to deduce an expression for the required time t , but Fitts did not do that. He only wanted to describe the empirical data in a consistent way.

The appearance of Fitts' law is historically situated in a time when academic psychology (in the US) was on its way back from the behaviourist dark ages. The appearance of the computer had made it possible to build testable models of human cognition. The channel idea has been central in the sciences of human cognition and performance. In the behaviourist version, the channel degenerates to a black box, but the basic view of the relation between subject and object remains. In human factors engineering, human-machine relations are seen as a circular composition of channels:
 Machine operation → displays → sensing → data processing → controlling → controls → machine operations → ...

Additive Models

From the viewpoint of experimental psychology, Fitts' law is an unquestionable fact, due to the overwhelming amount of empirical evidence. When we move outside the laboratory, however, this fact appears to be questionable. The implicit assumption underlying Fitts' law is that it is possible to decompose human performance into basic tasks and add up the times for all these tasks to get the total performance time. This strategy, known as additive models, is an old idea and has been questioned for at least the last hundred years [9]. Simplistically stated, the assumption underlying additive models is that e.g. a mouse operation is the same thing no matter what context it is performed in. In HCI additive models have been widely used (e.g. the GOMS- and keystroke level models by Card et al. [4])

An experiment reported in [9] shows that additive models cannot describe even simple and controllable tasks. The subjects were solving so-called “Sterzinger lines”, i.e. nonsense lines of letters and spaces. The task was to step from space to space, using the arrow keys on the computer console, until a space separating two equal letters was reached and then to indicate that by pressing the arrow-up key. In contrast to the studies by Card et al. [4], Gediga and Greif were able to monitor the performance of single keystrokes and thus discover that, although the total performance time could be described by an additive model, the time taken to perform the single key-press changed as the subject moved the cursor through the line.

The significance of these result with respect to Fitts’ law is that if it is impossible to build additive models, then we may expect that the parameters in Fitts’ law change according to changes in the (micro) context. Furthermore, the relevance of Fitts’ law as a prescriptive tool depends on the general validity of additive performance models.

The “Sterzinger line” experiments can be seen as a critique of additive models from within experimental psychology showing that the relation between the human being and the surrounding world is dialectical and not mechanical — while the subject changes the object, the object also acts on the subject. There is no stable engine inside the subject. A more fundamental critique can be made by questioning the validity of laboratory experiments as a source of design relevant knowledge. Chapanis [5] has pointed out that, due to the complexity of human beings, it is almost impossible to keep track of the variables. This critique can be radicalised from the point of view of activity theory by stating that the human being in the controlled environment of the laboratory, is fundamentally different from the human being in “the real world” [12]. Although some aspects of human performance are evident from a laboratory experimental point of view, these aspects may not exist outside the experimental setting.

Fitts' Law in HCI —some Examples

Fitts’ law studies expose great diversity in the way the law is used and the approach to the given (design) problem. Two general approaches can be identified. One tries to find general quantitative properties of the human motor system, constants of the human that are independent of the specific circumstances. In this group we find studies that have great resemblance to Fitts’ reciprocal tapping set-up, but no resemblance to any practical situation (e.g. [11]). These studies are based on the implicit assumption that Fitts’ law is part of a cognitive psychology that can form a universal framework for studies and design of HCI. At the other extreme, we find approaches more con-

cerned with specific issues of specific (types of) interfaces, and thus more realistic in the experimental set-up (e.g. [7]), here we see studies that use Fitts' law solely as a source of inspiration. We are dealing with two fundamentally different approaches to the possibilities of a HCI-science. Either you try to establish anthropometric laws based on a "theory" about human cognition, or you can, based on specific metaphors, study specific interface classes. The classic Fitts' law study in HCI by Card, English, and Burr [2] comparing the performance of various devices in text selection, can belong to both categories. Some see these experiments as the determination of I_p in the mouse version of Fitts' law, others see the studies as a concrete, although reductionist, investigation of specific input devices. I prefer the latter interpretation.

Gillan et al. [7] report that Fitts' law can account for some of the performance variations in text selection with a mouse, but that other variations can neither be predicted nor described by Fitts' law. Two cases that theoretically should have the same I_p were examined, nevertheless the two I_p appeared to be different. Gillan et al. conclude that a general theory of mouse performance in a direct manipulation interface has to include parameters for aspects not covered by Fitts law, e.g. cognitive processes and user strategies. They point out that development of design oriented metrics has to be based upon detailed investigations of what the user does in concrete situations. Thus they give up the efforts to find the parameters in the mouse version of Fitts' law, and view their studies as dealing with some concrete properties of direct manipulation interfaces.

The applicability of Fitts' law in a practical design task is illustrated by an example regarding the placement of "soft buttons" in a hypertext browser screen layout [10]. Fitts' law was utilised to minimise the time required for mouse operation.

We probably saved tens of milliseconds per 5-minute browse. This really is not bad, as such things go; it's often not done in commercial systems, and is economical worthwhile in our expected applications, which have large multipliers. I think saving small fractions of a second by optimal button placement is probably a good illustration of the real but limited impact that traditional psychological theory can have if diligently applied. [10, p.65].

This indicates that artefacts like Fitts' law can be used to solve specific isolated design problems, whereas they are almost useless as general perspectives on human-computer interaction.

Laboratory experiments by Gediga and Wolff [8] confirm that target size influence movement time in "mousing", at the same time as the quantitative contents of Fitts' law are considered too unpredictable to be applied in design. They suggest a distinction between Fitts' law and a Fitts' effect that

merely states that movement time is inversely proportional to target size, and proportional to movement length. They say that only the latter is relevant in HCI. This seems to preclude Fitts' law from being a part of the world of "task analysis, approximation and calculation".

Fitts' Law as a Design Artefact

Fitts made sense of a "chaotic" world by constructing a consistent predictive scheme. Although this scheme predicts and describes empirical data it provides no suggestions for an understanding of the observed phenomenon and its relations to its surroundings. Thus Fitts' law can be seen as a detached predictive metaphor. Fitts' law is basically a performance model in line with the time and motion study tradition founded by Taylor and Gilbreth. Together with this tradition it tends to reduce design of work environments, e.g. computer artefacts, to a matter of economical optimisation. No matter how much it is claimed that Fitts law is merely a useful metaphor, it will make us perceive the human being as a channel. The danger is that viewing the human being as a channel will make us treat her as a mechanical device. The significance of these basic assumptions about the human psyche depends on the part Fitts' law plays in the game of design.

The above examples show at least three different roles. In the HCI psychology by Card et al. [4], Fitts' law was part of the *world view*. To play this role, Fitts' law must be placed in a context where human beings are seen as mechanical devices, i.e. cognitive science. Landauer [10] used Fitts' law as a *tool* for specific calculations in the design process, without adopting cognitive science as his main perspective on HCI. Gillan et al. [7] used Fitts' law as a *metaphor* for research on specific aspects of acting with a mouse. In design this amounts to the use of the Fitts' effect as a thinking tool [8]. The borders between these three roles are not clear-cut. By thinking about interface problems in Fitts' law terms (the metaphor role), other views are excluded and one is led towards mechanistic reduction. The use of a specific performance calculation serves as a thinking tool, too. Theories always play different roles at the same time.

In Newell's implicit dichotomy between true and applicable theory the metaphor role could be added as a third distinction. I would rather prefer to view the three above roles as modes of acting with and developing understanding of the world. The methodological pragmatism expressed by Newell rejects that value statements can have any relevance in the real world of design, and that differences in the overall understanding of the use of computer artefacts can have any practical significance. A simple notion of theories as design artefacts, based on this view, would state that theories are

tools for prediction, calculation, and generation of visions; and that sometimes they work and sometimes they do not. Furthermore, such an idea would claim that the only valid world view should be the collection of tools for calculation, or performance models, just like the Model Human Processor [4].

In contrast to this, I will claim that value statements must be the basis for every science no matter whether they are implicit or explicit. Performance models like Fitts' law are based on specific (implicit) assumptions about the human being and her relation to her surroundings. By applying such models as tools in design we will automatically share this world view, unless we strongly specify another one. We can not avoid this ontological discussion. Our implicit or explicit choice of world view is also a choice of the world in which we want to live; disinterested sciences do not exist [1].

The absence of value statements in the methodological pragmatism of Newell [13] and Card et al. [3, 4] leads to either a position saying that any statement is valid if you like it to be so (i.e. relativism), or a position saying that only statements that can be inspected are valid in science (i.e. logical positivism). In the latter case, the scientific method is installed as a substitute/proxy/go-between for the assessment of theories, as it is seen in the idea that the psychology of the human-computer interface should be a *hard* science [14].

When designers build specific computer systems they use what they have and what they know, no matter how incompatible from a theoretical point of view. Current social- and cognitive science tend to misunderstand the strengths of science and just collect everything that seems to be right together. Scientific theories are not one-to-one reflections of the world, but artefacts mediating understanding of, and action in the world, through reduction. By stuffing everything together, nothing interesting about the world will appear, powerful theories have to be based on cruel reductions.

Conclusion

The fact that cognitive science is able to predict and describe many phenomena relating to HCI, should not necessarily lead to the conclusion that cognitive science must be (part of) the scientific framework for HCI. I still agree with Card, Moran and Newell that HCI design might benefit from a tighter connection between science and design, but as the use of Fitts' law in HCI indicates it is not likely that mechanistic psychology will form a fruitful basis for this connection. A pragmatic science of HCI will have to take into account the context of the use of computer artefacts, and the context of design of computer artefacts as well as the relation between science and design.

In a way it is both too optimistic and too pessimistic to state that: “Scientific models do not eliminate the design problem, but only help the designer control the different aspects” [3]. Of course, no science will ever be able to see into, or build the future. Human beings are fundamentally contingent, one is never sure of their next moves, and thus science will never fully control any aspects of the interface. On the other hand, a radical pragmatic science of HCI, a science not based on ideal natural science, can yield design-knowledge and understanding that goes beyond technical control. Such a radical pragmatic science of HCI will necessarily be based on dialectical, as opposed to mechanical materialism.

Acknowledgements

Thanks to Susanne Bødker, Randy Trigg, Morten Kyng, Kim Halskov Madsen, Preben Mogensen, Anders Mørch, and the anonymous reviewers for discussions, comments, and encouragement; and to Susanne Brøndberg and Janne Damgaard for last minute proof-reading.

References

1. Bertelsen, Olav W.: Når Uret Blir Ret... Et litteraturstudium til belysning af muligheden for teori i human-computer Interaction. [A literature study eliciting the possibilities of theory in HCI], Unpublished masters thesis, Aarhus, 1993.
2. Card, S.K.; English, W.K. and Burr, B.J.: Evaluation of mouse, rate-controlled isometric joystick, step keys, and text keys for text selection on a CRT, in *Ergonomics* vol.21, pp.601-613, 1978.
3. Card, Stuart K., Thomas P. Moran and Allen Newell: The Keystroke-Level Model for User Performance Time with Interactive Systems, in *Communications of the ACM*, vol. 23 pp.396-410, 1980.
4. Card, Stuart K., Thomas P. Moran and Allen Newell: *The Psychology of Human-Computer Interaction*, Hillsdale NJ, 1983.
5. Chapanis, Alphonse: The Relevance of Laboratory Studies to Practical Situations, in *Ergonomics*, vol. 10 pp.557-577, 1967.
6. Fitts, Paul M.: The information capacity of the human motor system in controlling the amplitude of movement in *Journal of Experimental Psychology* vol. 47, no. 6, 1954.

7. Gillan, D.J.; Holden, K.; Adam, S.; Rudisill, M. and Magee, L.: How does Fitts' Law Fit Pointing and Dragging?, in *Proceeding of CHI'90 Conference on Human Factors in Computing Systems* (pp.227-234), New York, 1990.
8. Gediga, Günter and Wolff, Peter: On the applicability of three basic laws to Human-Computer Interaction, *MBQ* 11/89, Osnabrück, 1989.
9. Greif, Siegfried and Gediga, Günter: A Critique and Empirical Investigation of the "One-Best-Way-Models" in Human-Computer Interaction, in Frese, Ulich and Dzida (eds.), *Psychological Issues of Human-Computer Interaction in the Work Place*, Amsterdam, 1987.
10. Landauer, Thomas K.: Let's get real: a position paper on the role of cognitive psychology in the design of humanly useful and usable systems, in Carroll, J.M.,(ed.), *Designing Interaction: Psychology at the Human-Computer Interface*, Cambridge, 1991.
11. MacKenzie, I. Scott; Sellen, Abigail and Buxton, William: A Comparison of Input Devices in Elemental Pointing and Dragging Tasks in *Proceedings of CHI, 1991*, pp.161-166, ACM, New York, 1991.
12. Mammen, Jens: Menneskets Bevidsthed [Human consciousness], in Fenger and Jørgensen (eds.), *Skabelse, udvikling, samfund*, pp.73-81, 271 Aarhus, 1985.
13. Newell, Allen: *Unified Theories of Cognition*, Cambridge Ma.,1990.
14. Newell, Allen and Card, Stuart K.: The Prospects for Psychological Science in Human-Computer Interaction, in *Human Computer Interaction* 1985 vol.1, pp.209-242.
15. Shannon, Claude E.: *The mathematical theory of communication*, Illinois, 1949.

Third paper

Supporting the Development of Transparent Interaction

Co-authored with Jakob E. Bardram

Abstract: Transparency has been seen as a significant aspect of successful human-computer interaction. In this paper we investigate this concept from the point of view of activity theory. We show that transparency cannot be understood as a static feature of the interface, but that the crucial point in achieving transparent interaction is the ongoing development of unconscious operations, embedded in the process of use. We suggest that the process of deliberate formation and refinement of operations during the course of interaction, is supported by setting conditions for the creation of a zone of proximal development in the interface.

Introduction

Better interfaces is a goal shared by most people in the HCI community, but it is harder to agree on what constitutes a good interface; and to what extent a scientific foundation for design is necessary. In the visionary contribution to a scientific foundation of HCI by Card, Moran, & Newell (1983; Newell & Card 1985), the criteria for good interaction were mostly of quantitative nature; maximum speed, minimum errors, fast learning, and other measurable quantities. The degree to which such criteria are met is easily assessed in an objective manner; unfortunately, many important usability aspects are not covered by such metrics.

A recent report sponsored by the US National Science Foundation claims that “The aim of HCI research should be to understand the principles behind what is necessary for transparent interaction.” (Strong 1994, p.19),

thus suggesting that transparency could be a measure for good interface design. It has been recognised by many in the HCI community (e.g. Hutchins et al. 1986; Laurel 1986, Winograd & Flores 1986; Shneiderman 1992) that people want to think about the job they are doing and not about the computer artefact they are using; but it is hard to find substantial explanations of the concept of transparency.

In specific situations it may be easy to see whether a computer artefact appears to be transparent to the user. Experienced Emacs users interact transparently with the editor although it appeared completely opaque at their first encounter with it. Emacs is a very powerful and transparent tool for many programmers, but it will probably be hard to persuade writers to use it. On the other hand, it is not likely that an artefact like MS-Word that many writers interact transparently with, can become a transparent tool for programming. This small example illustrates two things about transparency. Firstly, that transparent interaction is not a property of the interface by itself, but a quality of the use activity; and secondly, that transparent interaction is developed by the user during interaction.

In this paper we discuss the concept of transparency from the point of view of activity theory, and we try to identify how transparency is developed during use, and how this ongoing process can be supported by conditions in the interface.

HCI and the information processing paradigm

From the very beginning two distinct tendencies have coexisted in the field of HCI. One side mainly motivated by a practically oriented concern for the usability of computer artefacts; the other side more oriented towards the theoretical foundation of HCI. Norman's (1981) classical analysis of UNIX can be seen as an instance of the first, Card, Moran and Newell (op. cit.) as an instance of the later. On both sides information processing is the common framework; thus, understanding the user is often referred to as, "understanding the limitations of human information processing" (Strong op. cit., p.5). The predominant tendency in the field has been pragmatic and inclusive; a book like *User Centred Systems Design* (Norman and Draper (eds.) 1986), is an example of this, although often referred to as cognitive science.

With respect to aspects of interaction that can be regarded stable, or constant over time, the information processing based tradition has established valuable knowledge concerning cognitive abilities of generalised users like

the “expert user”, ranging from studies of performance at a keystroke level (Card, Moran & Newell, 1983) to high level concepts like direct engagement (Hutchins et. al. 1986). Due to the explicit or implicit idea of information processing, the recurrent problem is that the HCI community tends to see their task as one of fitting two separated, more or less stable engines to each other.

The most simple information processing based models like GOMS (Card et al. op cit.) are extremely additive, and any notion of context is absent; typing an “H” is basically the same thing no matter whether it is the first letter in “Hoovercraft” or it is pressed to get on-line help. It is impossible to explain how the same text editor can be different things in different situations — how the use qualities of computer artefacts develop in the process of use. Within such cognitive models accounts on learning are mostly of a quantitative nature, the power law of practice (Snoddy 1926; Card et al. 1983) describing how the speed of a human act develops as a function of repetition, being one of the most prominent examples.

In more elaborate models like production systems as ACT* (Anderson 1983) and SOAR (Newell 1990) the additivity is not obvious, and the models explicitly try to deal with more qualitative aspects of learning and development. The problem remains that the way human beings relate to their surroundings is understood as a kind of problem solving, or search in a problem space; and skill acquisition is thus understood in terms of chunking or other “shortcuts” in the problem-space-search.

Such models can describe how skills in operating an advanced piece of technology can become automatic. The way users relate to their surroundings is however still seen as an exchange of information between distinct entities; thus the subject-object relation is seen as secondary compared to the internal and isolated processing of information. By drawing this distinction between the users as information processing engines and their surrounding, the information processing paradigm precludes itself from understanding how the objective conditions of the world and their cognitive representations are inseparable parts of cognition. This separation is problematic in understanding transparency because it separates the description of development from the description of the use-activity in which development is embedded.

Although it is possible to deal with some of the dynamic aspects of the relation between users and artefacts within information processing psychology, such accounts tend to become exorbitantly complicated. We will not enter into a philosophical discussion about the general validity of different approaches to human cognition; after all, no objective or value-free methods for settling such abstract controversies exist; the choice of scientific foundation can never be legitimated by an external norm or method; we have to take the responsibility for the way we choose to understand and act in the world in which we live (Feyerabend 1975, Habermas 1972). We will, however, show

that a theoretical foundation for HCI, based on the dialectical nature, and fundamental situatedness of the user-computer relation, can yield a more useful understanding of transparency than an approach dealing with the dynamic relations as an exception. By changing the basic assumptions from the mechanical materialism of cognitive science to the dialectical materialism of activity theory, it becomes possible to understand how transparency is tied to the concrete context of use, and how transparency develops within this context.

The structure and development of human activity

Human activity has three fundamental characteristics; firstly, it is directed towards a material or ideal *object* which distinguishes one activity from another; secondly, it is *mediated* by artefacts (tools, language etc.); and thirdly, it is social within a *culture* (Vygotsky 1978). Computer artefacts, like all other artefacts, in this way mediate human activity within a practice. By acting in the world, human beings meet the objective world, which is experienced through the activity. Thus, human knowledge about the world is *reflection* obtained through activity, constituting the basis for expectations, and desires about activities in this world.

Human activity can be described as a hierarchy with three levels: *activities* realised through chains of *actions*, which are carried out through *operations*. At each of these levels the objective world is reflected through the activity. Human activity is always directed toward a material or ideal object satisfying a need. The subject's reflection of, and expectation to, this object characterises the *motive* of the activity. Human activity is carried out through actions, realising objective *results*. These actions are controlled by the subjects conscious *goals*. Goals are the reflection of the objective results of the action. Actions are realised through a series of operations; each determined by the concrete physical *conditions* of the action. These operations are performed without thinking consciously but are oriented in the world by a non-conscious *orienting basis* of the operation. This orienting basis is established through experience with the concrete material conditions for the operation, and is a system of expectations about the execution of each operation controlling the operation, in the process of the activity (Leontjev 1978).

Human work is collaboration mediated by different artefacts in order to realise common objectives. The overall objective of the organisation is divided into various activities, each having its own object and performed by different persons. Each person takes part in the activity of getting the whole organi-

sation to work. The collaborative way of doing things and the division of work between workers in the organisation form a working culture — a practice. Means of dividing work, norms, and language can all be seen as artefacts mediating this practice: they are made by human beings and mediate the relations between human beings and their material and social surroundings. A secretary may need to write a letter to remind Mr. Smith that he is late with his payments and is therefore motivated to initiate actions to achieve this. This activity is carried out through actions, realising objective results: to write a letter the secretary must find an address, formulate the letter, get a signature, etc. Activity and actions cannot be reduced to each other: Various actions can result in a letter — handwriting, typing or using a word-processor — all different actions, mediated by different artefacts, obtaining different results but still fulfilling the same objective of writing a letter. On the other hand, the same action can be a part of realising different activities. Printing a letter on the laser-writer can both be part of the paper-archive-maintenance activity and the send-a-reminder-to-Jack-Smith activity. The analytical level of actions describes the intention of an activity — what results there are or should be obtained. Operations describe the operational level of the activity — how the action is realised, determined by the actual physical conditions of the action. The layout of the keyboard, the word processor's interface, etc. determine how the secretary uses the computer to realise the goal of writing the letter. The non-conscious orienting basis of the operation enables the skilled secretary to write the letter without directing attention to the layout of the keyboard.

Within the framework of activity theory, transparent interaction can be defined as handling the computer through operations. In this way the computer artefact mediates the activity by allowing users to perform intended actions directed on the object (inside or outside the computer) through the interface. At a given moment the user has a certain repertoire of operations; the only way interaction can be transparent is if conditions of the interface triggers operations in this repertoire (Bødker 1991).

Human activity is not stable but is transformed constantly. An activity can lose its motive and become an action with the former motive as goal. An action can lose its conscious goal and be subsumed under another action as an operation. This process of *automation* happens through practice. The reverse process of *conceptualisation* happens when the conditions for the operation force users to think consciously about what they are doing. This happens in a breakdown situation (Winograd & Flores 1986), where the physical conditions for the operation do not match the operations orienting basis, or it happens due to deliberate focus shift from the goal of the action to the conditions for the operation (Bødker 1991). Such breakdown situations or intended focus shifts where the computer becomes object of conscious examination interrupt the transparency.

The user's repertoire of operations is constantly developing, allowing the computer artefact to become an effective support in a work situation. The operations used by skilled Emacs users were not present when they started to use the editor, they were established during use. At first the act of marking, copying and pasting some text is done by consciously taking one step at a time, deliberately making sure that the different acts obtain the desired result. The conditions of the editor for placing the cursor, marking the text, etc., are incorporated into the orienting basis. Through practice the different parts of the action become automatic and finally the total action is automated into an operation, which can be executed without conscious reflection. But if for instance the conditions for an operation change due to changing mode of the editor, the user may experience a breakdown. This conceptualisation makes the relation between the conditions and the orienting basis of the operation an object for conscious reflection, in order to fix the discrepancy.

Development of operation through automatisatisation of actions

The main mechanism for the development of a repertoire of operations is *automatisatisation* of actions to operations, where the conscious guidance of the action toward a goal has stopped. This does not mean that the user does not perceive the conditions for the operation, instead the operation is oriented by an unconscious perception of these conditions.

The efficiency of the transformation of actions into operations is determined by the formation of the actions. According to Gal'perin (1969) there are three basic parameters characterising the quality of an action. Its *generality*, its level of *abbreviation* and the extent to which the action is *mastered*.

The generality of an action is determined by the orienting basis of the action. To generalise an action is to distinguish the cues of the objective conditions, which are necessary for the fulfilment of the action. We can differentiate various forms of orienting bases for an action, each determined by the learning process in which the action is developed. Trial and error learning results in an orienting basis solely directed towards the actual action. Such an orienting cannot be transferred to other actions and represents the lowest generality. The most general orienting basis is established through analytic learning of general methods of analysing the conditions of the action. These methods of analysis are incorporated into the orienting basis of the operation as methods for recovering from breakdown situations permitting the user to isolate the structure of the conditions for later recognition or rep-

etition. Transparency of interaction is closely connected to the generality of the orienting basis. Abbreviation is the process of optimising the action by skipping unnecessary operations and replacing difficult operations with simpler ones, as a result of possibilities in their conditions and knowledge of their result.

When generalising an action users come to know the full scope of conditions in the surroundings that the action matches; and the full scope of potential desirable goals that the action can fulfil. When generalising the action of invoking the UNIX “cat”-command users begin to use the command not only for concatenating files but also for printing files on the screen, feeding them into pipes etc. To do this users need a general orienting basis. Similarly, abbreviation depends on the creation of the orienting basis. Removing a file from a UNIX file system may, for the “inexperienced” user, involve the execution of one or more “ls”-commands (list files in the current directory) to make sure what is going on, in addition to the “rm <filename>”-command (remove the file) . When the user gets to know what is going on the remove action is abbreviated by skipping the “ls”-commands. Similarly, using a keyboard short-cut on a window based interface can be seen as an abbreviation of the more complex mouse-operated menu selection.

Finally mastering the new action is the ability to independently perform the action with new materials in unanticipated contexts, in order to obtain a new result. If users learn to drag and drop objects into the Macintosh trash can the mastering of the drag and drop action implies that users, on their own, are able to drag and drop objects into folders etc. Users do not come to master an action by watching others, but only by practising the operation themselves.

The process of establishing the action is a conscious process in which the user forms an orienting basis according to the material conditions for the action. Once the action has attained a certain level of generalisation, abbreviation, and mastery, it can be automated as an operation. Once the action has been automated, the process of enhancement has stopped. It is only possible to develop the operation by conceptualising it back to an action, consciously reforming this action and then automating the changed action.

When development is embedded in the daily work, one problem is that the process of refining a new action requires time. In a given work situation users may not be interested in mastering brief and general operations; the primary motive is to get the job done (quickly). In such situations learning is biased toward trial-and-error learning which establishes infertile operations only adjusted to the conditions for this specific situation.

Setting the conditions for development of transparent interaction

Mediated and social human activity builds up the human world as artefacts embodying human capabilities, developed through socio-historical practice. Human work is characterised by the collaborative production of artefacts, each made with the purpose of mediating a certain activity. This activity is therefore objectified (or crystallised) into these artefacts, and through use, the artefacts are again modified and shaped to meet the evolving human needs. Window based interfaces are artefacts, which objectify human experiences with computer interaction. An artefact like the Macintosh is a material objectification of a large number of experiences obtained during several years, ranging from the invention of the mouse by Doug Engelbart over the pioneering steps of the Xerox Star to the latest work on intelligent agents.

To assimilate the human world as made by artefact embodying human experiences, the human being must perform overt activity with the artefacts; activity adequate to the human activity embodied in these artefacts. Furthermore, this activity should be socially mediated by other people within the culture. It is a fundamental notion of activity theory that human beings, guided by other human beings, gradually assimilate the human world by reproducing the activity embodied in the artefacts. The child learns to use a spoon, not on its own, but with the help of the mother. By imitating the activity that was originally crystallised into the spoon the child tries to master it as a human artefact. Similarly, users rely heavily on colleagues for developing their skills and learning about the computer, especially novice users (Bannon 1986). Vygotsky captures this social mediation of human cognitive development in the concepts of “the zone of proximal development”:

“It is the distance between the actual developmental level as determined by independent problem solving and the level of potential development as determined through problem solving under adult guidance or in collaboration with more capable peers” —Vygotsky 1978, p. 86

The concept of a zone of proximal development describes how the mediated activity develops dialectically through contradiction between what users can do with the tool today, their *actual competence*, and what they want to be able to do in a near future, their *potential competence*.

The actual competence of the user forms the basis for development; it is the user’s species specific, cultural and individual experiences — what the user has learned up to this point in time. Traditionally HCI has looked at the actual competence of the general user. Direct manipulation, for instance, captures some of the human experience as a specie and enables a “natural” interaction: human beings manipulate objects by grabbing, dragging, lifting

them; the shape of objects is perceived according to the normal direction of light from the sun, etc. A more significant part of the user's actual competence is the "computer-cultural experience"; i.e. culturally established ways of interacting with computers. Window based operating systems provides a type of interaction, which is becoming gradually more culturally objectified through the common use of windows, mouse, buttons, scroll bars, pull-down menus, etc. Consistency in the use of a window based interaction makes it possible to transfer experience from one system to another. Hence, traditional HCI has searched for general principles of how computer artefacts can conform to the cultural- and species specific experiences of users. The individual experience, however, is impossible to grasp in general terms, and thus widely ignored by the field of HCI. It is, nevertheless, a fundamental part of the actual competence that has to be considered.

Based on the actual competencies the user must be able to look ahead of the actual zone of mastery and create a zone of proximal development. Gal'perin (1969) points out that children are unable to perform a task independently, if they have no complete orienting basis. Thus, learning of a new task requires that users are given an initial familiarity with the task; otherwise learning will be reduced to 'blind' trial and error. Human beings can deliberately create or seek special conditions with the purpose of learning, engaging themselves in an exploratory learning activity. According to Bol (1984), Vygotsky's concept of a zone of proximal development has a double function with respect to cognitive development. Firstly, learning activity creates a zone of proximal development and, secondly, learning is collaboration with "more capable peers". To develop transparent human-computer interaction we need to establish the right conditions for this learning activity — be it with or without "more capable peers". In this paper we use the concept of the zone of proximal development in the metaphorical sense as a zone the user is able to set, supported by the conditions built into the interface.

Three issues concerning the development of transparent interaction emerge from the general understanding of the development of human activity: Firstly, development only takes place *within* a purposeful use-activity. Establishing competencies as a repertoire of operations must correspond to a certain goal and motive. Without establishing this dialectic relation between learning and use, development is impossible. Secondly, the user must be able to *reflect the potential mastering* of the computer, and how that level of the use-activity can be reached. Learning of a new task requires an initial familiarity with it in order to "look into" the zone of proximal development — otherwise learning will be 'blind' trial and error. Finally, the user must be provided with conditions for consciously *establishing* actions that are general, abbreviated and mastered to a certain degree; and for automating these new action into new operations through practice. In this way, the orienting basis of the coming operation is consciously formed and optimised according to the material conditions.

These issues concern general aspects of users' surroundings that should be met to enable the development of new operations. For example: the abbreviation of mouse operations into keyboard short-cuts depends on conditions set at the interface by indicating the short-cuts next to the command names in the menu instead of placing a description of all shortcuts in a manual, forcing the user to interrupt the work-activity and go look for the manual. Similarly, the mastering of brief and general operations requires time, but this may be inhibited by conditions of the work environment such as stress, causing the user to develop infertile operations. Development of interaction is not isolate from "the rest of the world", and the concept of human activity emphasises this connectedness.

Designing for transparency

It is impossible to identify general "transparent features" to be included in all user interfaces and list these as general applicable guidelines for "the design of transparent computer applications". The conditions in the interface that are necessary for the development of transparent interaction depend on the users and their jobs, and the way the introduction of the new computer artefact intervenes and changes this relation. This means that in the design of computer applications we have to rely on participatory design methods like prototyping and organisational games (cf. Greenbaum & Kyng (eds.), 1991), and a broad range of more traditional user testing techniques in order to ensure the presence of conditions for development and transparency, in the interface. For example; Frederiksen et al. (in press) has showed that inconsistency in scrolling direction on word-processors and scrolling direction in molecule-modelling tools yields the most transparent interaction for chemistry students; questioning the universality of the common belief that the universal direction of scroll bars should be "up moves the window up".

The challenge in designing for transparency is to ensure that the user interface provides the conditions for creating a zone of proximal development of operations. That is: designing the interface to mediate the initiation of and learning through exploratory actions, subsumed under the overall use-activity; that the interface enables the user to look ahead of the zone of actual development, and to explore conditions for obtaining different new results; and through this exploration support the development of new ways of realising the object of the activity, without entering another separately motivated activity.

While it is impossible to give general guidelines concerning specific features of "transparent interfaces"; it is possible to elaborate on the three issues

concerning the development of transparent interaction stated above, in order to yield a more design oriented understanding of the conditions for development of transparency. That is, supporting development *in use*, ensuring a certain degree of *initial familiarity* and setting conditions for the *formation of new operations*.

Development in use

The first issue deals with the necessity to support the development of the use-activity within the use-activity. In general, we cannot completely avoid separately motivated learning activities like reading introductory documentation, attending guided tours, etc. If users do not know what a mouse is and how to operate it, they will not be able to do anything with, e.g. a Macintosh computer. In such situations the only solution is a more explicit socially mediated way of learning; via personal guidance, or by written advises on the screen or in a manual. With respect to transparent interaction the interesting issue, however, is how support for development can be an integrated part of day to day use, when a use situation is established.

The main problem with UNIX and Emacs is that although the possibilities for transparent interaction exist for programmers the actualisation of these possibilities is difficult due to the absence of support for learning within the use activity. To learn something about the systems it is necessary to consult manuals or colleagues, which possess more knowledge about the system, thus separating learning from doing. The use of on-line help systems, especially context sensitive help-systems, can be seen as a way of supporting development in use. However, an on-line help system can also drag the user away from the use-activity by being too difficult to use. Furthermore, the entire concept of on-line help is based on the idea of separating learning from doing by providing users with external descriptions instead of supporting development directly by features of the interface.

Initial familiarity

The second issue deals with the user's ability to reflect the potential mastering of the computer — to look into a zone of proximal development. The interface must provide users with an idea or initial familiarity of what they might be able to do in near future, and thereby encourage them to initiate exploratory learning actions. The concept of affordance (Gibson 1979) captures this at a very low level of interaction; the properties of an object give the user an idea of its potential use. On a more overall level the use of metaphors can give the user an idea of potential use of the artefact by providing an analogy between something known and concrete, and something unknown and abstract. In the same manner, consistency between applica-

tions, e.g. the organisation of menus on the Macintosh, supports the transfer of skills from one application into initial familiarity with another.

Because learning (as opposed to mere adapting) requires conscious reflection of the conditions for the future operation, an exploratory learning action will interrupt the transparent flow of operations. This issue of providing users with an initial familiarity, therefore implies a trade-off between inviting (or even forcing) them to initiate learning; and allowing them to continue the accomplishment of their jobs. Therefore we propose that the emphasis must be on the clues built into the interface; this could be called graceful tutoring.

The initial familiarity encouraging exploration must not inhibit the ongoing use of the artefact but, on the other hand, the interface must exhibit the possibilities of the artefact in order to give users ideas of its future use. The way this trade-off has to be solved cannot be answered in general. It very much depends on who is going to use it, their use activity, the need the artefact tries to solve, the motive for the users to use it, the context of use, etc. A simple issue as whether it is sufficient to present the functionality of a window based application in pull down menus or whether it should be represented as buttons and icons, depends on the way the application is going to be used. If it is going to be used often, having buttons and icons all over the screen might be very annoying, especially if the space could be used for another application supporting the current activity. But, on the other hand, hiding functionality away in a pull down menu might prevent the casual user from learning about the different possibilities of the application. Thus, it is necessary to know how the artefact is going to be used in order to establish exactly how the interface should invite users to exploration.

The formation of new operations

The third issue deals with the formation of operations. Once exploratory actions have been initiated, the conditions provided by the interface determine how an operation is established. Thus the interface should set conditions that support the formation and mastering of actions that are generalised and brief. The essential aspect of the formation of new actions is the development of the orienting basis for the actions. Thus, designers should consider how the creation of a generalised orienting basis can be supported.

UNIX has highly general commands and a basic structure (pipe-lining etc.) that supports this generality, thus the operating system yields a possibility for highly generalised actions of using the commands. Furthermore, it has the potentials for abbreviated actions. UNIX does not, however, support the *development* of a generalised orienting basis for the actions of execution of these commands. Hence the general and brief usage of UNIX is seldom achieved through use.

Feedback from the artefact about results obtained in the course of mastering an exploratory action determines how the user learns about the artefact. The command based interaction of UNIX only supports trial-and-error learning because the feedback on commands consists only of prompting for a new command, indicating that the system has done something, or displaying uninformative error messages like “Command not found”. A more analytical learning of UNIX requires that users get to understand the structure of the operating system — the file system, pipes etc. The only obvious way for users to obtain this knowledge is to enter a separately motivated and detached activity of learning the basic concepts of the operating system.

To support the development of transparent interaction the designer must ensure that feedback from the artefact provides users with a general understanding of the artefact and its use. From the feedback and other cues users must be able to achieve an understanding of the full range of an action, the underlying assumptions, the different results that can be obtained, and how it can be performed in a brief way. This is to learn how the artefact mediates an activity directed toward an object and master the operation necessary for using the artefact in an activity. If the object of the interaction is to handle files and programs in an operating system like UNIX, feedback about the state of the file system (e.g. the directory structure, list of files in a directory etc.) is essential. Emacs supports feedback about matching parenthesis, indentation of nested loops, etc. and therefore mediates a transparent interaction toward the object of writing source code. Mastering an action is the ability to perform it in novel situations with unexpected materials; if users are supplied with sensible feedback and the possibility of reversing actions with unwanted results, the distance between exploring and using the artefact becomes smaller.

The designer supports the development of transparent interaction embedded in use through the lay-out of the artefact and the various kinds of feedback from the artefact. The overall arrangement and behaviour of items on a screen can provide cues to the underlying structure of the application and thereby support the development of transparency in use. How the initial familiarity and the feedback during use should be designed in a specific artefact, depends on the object of the interaction. Therefore, in order to design the way the artefact looks and responds during use we need knowledge about the structure of the activity that the computer artefact is intended to mediate and the object of this activity. Because knowledge about how a new artefact will mediate an activity cannot be obtained separated from the activity the designer needs to apply methods for designing the artefact in use. This calls for prototyping and other participatory design techniques.

Conclusion

The starting point of this paper was that HCI research based on information processing psychology is unable to comprehend the conditions for the widely accepted desire to design for transparency. This deficiency was identified as a result of the failure to recognise the dialectical nature of human beings' relations to their surroundings. The conceptual framework of activity theory made it possible to analyse the otherwise vague concept of transparency. This showed that transparency is not a static feature of the interface but an evolving qualitative aspect of the relation between human beings and computer artefacts. Transparency of interaction depends on the learning process that goes on synchronously with the process of use; thus, the important issue for the interface designer is to design the conditions for this development.

We have pointed at the need for a thorough reconsideration of the theoretical foundation for human-computer interaction. We do not reject the established pool of results from mainstream HCI research and practice, but we find that the present lack of a coherent theoretical foundation is preventing the field from taking the next great step forward. We have pointed at activity theory as a possible foundation for HCI. By doing this we do not suggest that activity theory is the only valid theoretical foundation or that it does not have limitations, but we think it is worthwhile to catch up on the growing interest in activity theory in the HCI community (e.g. Bødker 1991, Nardi 1995, Bannon & Bødker 1991, Bannon & Kuutti 1993).

The main conclusions with respect to design is that the inseparability of learning and doing should be taken into account by designing a curriculum for development integrated in the computer artefact. Development of transparent interaction can be supported by providing cues and feedback according to the structure of this object. This should be based on analysis of the object of the activity, that the artefact is supposed to mediate. The aspects of transparent interaction discussed can serve as points to consider during the design process. We hope that our theoretical contribution can be a source of inspiration for designers.

Acknowledgements

We want to thank Annette Aboulafia, Susanne Bødker, Tania Funston, Kim Halskov Madsen, Niels Jacobsen and the anonymous reviewers for comments on this paper.

References

- Anderson, John R. (1983), *The Architecture of Cognition*, Cambridge MA: Harvard University Press.
- Bannon, L. (1986) Helping Users Help Each Other, in Norman & Draper (eds.), *User Centered System Design*, San Diego.
- Bannon, L. & Bødker, S. (1991) Beyond the Interface: Encountering Artifacts in Use, in Carroll, John M. (ed.) *Designing Interaction*.
- Bannon, L. J. & Kuutti, K. (1993) Searching for Unity among Diversity: Exploring the “Interface” Concept, in *Proceedings of INTERCHI 1993*.
- Bol, E. (1984) On the Development of Learning Activity. In Hedegaard, M., Hakkarainen, P. & Engeström, Y. (Eds.) *Learning and Teaching on a Scientific Basis*, Aarhus University, Institute of Psychology.
- Bødker, S. (1991) *Through the interface: a human activity approach to user interface design*, Hillsdale, N.J.
- Card, S. K.; Moran, T. P. & Newell, A. (1983): *The Psychology of Human-Computer Interaction*, Hillsdale NJ.
- Feyerabend, P. (1975), *Against Method*, London.
- Frederiksen, N., Grudin, J. & Laursen, B. (In press) Inseparability of Design and Use:
An Experimental Study of Design Consistency. To appear in *Proceedings of the Computer In Context Conference, Aarhus Denmark, August 1995*.
- Gal’perin, P. Y (1969) Stages in the Development of Mental Acts, in Cole & Maltzman (eds.) *A Handbook of Contemporary Soviet Psychology*, New York: Basic Books.
- Gibson, J. (1979) *The Ecological Approach to Visual Perception*, Houghton Mifflin, Boston.
- Greenbaum, J. & Kyng, M. (eds.) (1991), *Design at Work*, Hillsdale: LEA.
- Habermas, J. (1972) *Knowledge and Human Interests*, Boston: Beacon Press.
- Hutchins, E.L.; Hollan, J. D. & Norman D.A. (1986), Direct manipulation interfaces. in Norman & Draper (eds.), *User Centered System Design*, San Diego.
- Laurel, B. (1986), Interface as Mimesis, in Norman & Draper (eds.), *User Centered System Design*, San Diego.
- Leontjev, A. N. (1978) *Activity, consciousness, and personality*, Engelwood Cliffs NJ: Prentice Hall.

- Leontjev, A. N. (1981) *Problems of the development of the mind*, Moscow: Progress
- Nardi, B. (ed.) (1995) *Context and Consciousness: Activity Theory and Human Computer Interaction*, Cambridge: MIT Press.
- Newell, A. & Card, S. K. (1985): The Prospects for Psychological Science in Human-Computer Interaction, in *Human Computer Interaction 1985* vol. 1, pp. 209-242.
- Newell, Allen (1990), *Unified Theories of Cognition*, Cambridge Ma.
- Norman, D. A. (1981) The Trouble with UNIX, in *Datamation* vol. 27 no 7.
- Norman, D. A. & Draper, S. (eds.) (1986), *User Centered System Design.*, San Diego.
- Shneiderman, B. (1992) *Designing the user interface*, Reading MA.
- Snoddy, G. S. (1926), Learning and Stability, in *The Journal of Applied Psychology* vol. 10 pp. 1-36.
- Strong, G. W. (ed.) (1994), *New Directions in Human-Computer Interaction Education, Research, and Practice*, Drexel.
- Vygotsky, L.(1978) *Mind in society: The development of higher mental processes*, Cambridge, MA.
- Winograd, T. & Flores, F. (1986), *Understanding Computers and Cognition -- a new foundation for design*, Norwood, N.J: Ablex.

Fourth paper

Contradictions in the Festival Project: Activity systems, obstacles, and dynamic forces in design

Abstract: The paper describes a research project which conducted participatory design in a non-profit organisation based on the work of volunteers. The project is analysed by identifying contradictions in the course of the project, and then fitting these contradictions into the activity theory based, developmental work research framework. It is concluded that the perspectives of contradiction and activity theory are valuable tools for making sense of design projects, and challenges for the further development of activity theory related to heterogeneity issues are identified.

Introduction

In this paper a case story about a project involving a group of researchers and people from a non-profit organisation, the Festival, producing an annual music festival is outlined. As a design project the project was not a success, but since it can induce and inform general discussions and challenge existing methods for cooperative design, it was a clear success as a research project. In this paper the Festival project is analysed in terms of contradictions. Contradictions, mostly as obstacles for the design project, have been identified without a special theoretical perspective; subsequently these contradictions are classified as contradictions in and between activity systems (Engeström 1987). It is concluded that notions of economical contradiction are still a necessary basis in understanding and acting in the design of computer artefacts no matter how much such ideas seem to be out of vogue after the collapse of the Soviet Union and the demolition of the Berlin wall

The Festival Project

The project took place during the first half of 1995, and involved a group of researchers from a university, and a music Festival organisation. In the fall of 1994 The Festival decided that they needed an IT-strategy: an internal IT group was formed, and the Festival contacted the researchers to initiate a project eliciting the possible advantages of introducing IT in the planing and production of the festival. A member of the Sound and Light group (see below) who had IT experiences from another music festival started discussions internally in the Sound and Light group about ideas for computer support for pre-production, prior to the project start. Also, it was decided by the Festival organisation to buy a number of PCs and a local area network with a server and printers, in a configuration similar to the installation the head of the internal IT group was the administrator of in his daytime job, and a number of laptop PCs were given to key volunteers at “middle management” level in the organisation.

Participants in the project were different members of operation groups from The Festival, and a group of researchers, consisting of three senior participatory design researchers, and two apprentice participatory design researchers.

The Festival

The Festival is a non-profit organisation with the production of an annual music festival as its main objective. In 1995, the festival took place over 4 days, with concerts on 8 different stages, presenting a total of more than 140 different acts. The making of a music festival involves many different tasks: engaging the artists, establishing camping areas for the audience, selling tickets, selling food, controlling access to the festival site, informing the press, building the festival site, etc. The volunteers working in The Festival are organised in 35 operation groups; 150 are working throughout the year and in the time around the festival additional 2500 volunteers are enrolled. 9000 members of external organisations (e.g. boy scouts and sports clubs) are working during the festival. 10 people have a regular, paid job at The Festival.

The focus of the project was on technical production and pre-production, involving people from the Booking group, Sound and Light group, Transit group, Catering group, and the eight stage groups (Green, Red, etc.), each consisting of 3 to 5 persons. The Booking group is responsible for deciding which artists are going to play at the festival, and for negotiating the conditions and prices with the agents. The Sound and Light group is responsible for the technical side of the artists performances. They are responsible for

making arrangements with sub-contractors running the basic sound (PA) and light equipment on the individual stages, and for making sure that all the artists will have the requisites needed for their performances equipment-wise, including instrument amplifiers, piano tuners, and help with special effects. In addition, the Sound and Light group has a co-ordinating role in the pre-production. The Transit group is responsible for the transportation of the artists from airport to hotels to festival, etc., and the booking of hotel rooms. The Catering group is responsible for dressing rooms, meals, and other backstage facilities for the artists. The festival takes place at eight different stages. The stage groups are responsible for the production at the stages, including establishing the facilities at the backstage area, e.g. stage and production offices, stage hands, etc.

Pre-Production

During the spring, the head of the Sound and Light group is employed at The Festival to take care of the technical pre-production, and to distribute relevant information to other operation groups. The most important means of communication throughout this process is telefax, and to some extent telephone. Pre-production work is a kind of detective's work; when an artist is booked for the Festival, the normal situation is that the only information The Festival gets is the name of an agency somewhere. Thus the first difficult task in pre-production is to find somebody who actually knows something about the artist, and then to convince this person that The Festival needs up-to-date information as soon as possible. Pre-production work is complicated by several factors. People in the music business are always late with everything; it can be hard to make people understand that The Festival needs information in advance. Also, it is very important for especially the bigger artists to show off by demanding specific resources for their appearance at the Festival, and these demands then have to be negotiated in some way. Finally, information about the Festival program, and information about arrival times and hotels for the artists has to be treated confidentially. Program information has to be kept secret until it is given to the public, to maintain the advertising value; information about hotels and arrival, likewise to protect the artists during the booking negotiations and during the Festival (e.g. stars do not like to have their hotels invaded by fans and the press). The general understanding in the Sound and Light group is that the festival could be produced without pre-production, but that it then would be more chaotic. Thus the purpose of pre-production is to facilitate a smooth production with a relaxed and friendly atmosphere.

During pre-production, Sound and Light builds a band file, a plastic folder enclosing documents, for each performance. These files are kept in a matrix of cardboard boxes, with one column per stage and one row per day. The first document in the file would normally be the checklist (see: below). A sheet of

paper organised in the same way, with the total plan of performances - the play plan - is used both as a tool for locating the files in the cardboard boxes, and for recording central information about the specific performances, e.g. the state of the information gathering, and the need for special equipment. The play plan is always situated on the desk in the Sound and Light office; when someone calls on the phone, the Sound and Light person will look at the play plan, locate the artist in question, and examine the state of the pre-production for this performance; then the pre-production person will take the file in the cardboard box while continuing the discussion on the phone.

The checklist is a sheet of paper with pre-printed fields for information related to a specific artists performance at the festival. The checklist was originally invented in the Green stage group. This list contained fields for all the information that should be available or collected when an artist arrived at the backstage area at the Festival site. The checklist was filled in when the Sound and Light group handed information from the pre-production over to the stage groups, and it was later used when the artist arrived. Subsequently it was adopted by the Sound and Light group, and used during the pre-production process as the central overview of the individual artists. From 1993, the Sound and Light group produced a common checklist for all the stages, and filled in the available information about the artists before handing over the complete files to the stage groups. Thus the checklist had three functions: firstly, as a tool for the collection of information during pre-production, secondly, as a medium for forwarding information from the Sound and Light group to the stage groups, and thirdly, as a tool at the stages when receiving the artists and facilitating the performances.

The project

The first meeting between The Festival and the researchers took place in the middle of January, and in the beginning of February the researchers decided to engage in the project followed by the forming of an agreement for the project.

The first project meeting took place at the end of February; the Sound and Light group told the researchers about The Festival from their perspective, and about their work. The researchers demonstrated some of their own software as inspiration for the Sound and Light activists. The Sound and Light group had prepared for the meeting, by making two descriptions on paper, one describing the “flow of information” to and from the Sound and Light group during pre-production, the other a sketch of a database for pre-production represented as a screen layout.

During March, a series of interviews with two of the stage groups, the Catering group, the group responsible for access to the festival area, the Transit group, the Booking group, festival management, and a secretary,

were conducted. The first of April a workshop with Sound and Light, Transit, Catering, and the Yellow stage groups took place. After the workshop the researchers decided to use a database management system (hereafter The DBMS) to have a prototype ready before the big rush of the pre-production activities. During April, the researchers designed and implemented a first prototype of a system for Pre-production.

By mid-April, the Festival management became nervous about the project, fearing that too much information would flow too freely inside and outside the organisation, and therefore they dictated that the project could only continue with the Sound and Light group. As a consequence, the second and third planned workshops with the operation groups had to be cancelled. This breach of the original agreement made it difficult for the researchers to continue the work in a decent manner; in particular it became impossible to confront their understanding of the festival with the actual reality. A further complication during this period was that it was unclear if the Festival management would allocate a PC in the pre-production office.

During May, the first version of the prototype was installed at the Sound and Light office, and some of the existing data was entered into the system by the researchers. This version was never used by the Sound and Light group. At the end of May a revised (simplified) version of the prototype, more suited for the situation with the Sound and Light group as the only users, was installed. Because of expectations to the prototype, and because of difficulties induced by reorganisations of the Festival secretariat, Sound and Light had not made the checklists as they did it during the previous year's pre-production at this time. Some of the data was entered into the database, but most of it was only available on faxes, and ad hoc notes. The final result was low quality checklists delivered to the stage groups.

During Festival 95 in the last week of June, the researchers conducted field studies at the Festival-site; interviews with activists as well as observations of work.

On the 25th of August, the researchers sent the first version of a report on ways to improve the work of The Festival by the use of IT, to the internal IT group and Festival management; the report was also sent to the people who had been involved in the project or in other ways contributed to the report, for them to correct possible mistakes.

On the 5th of September the Festival management sent a letter to the researchers, stating that they saw the fact that the report was sent to the volunteers as a violation as the conditions for the project. Management stated that the researchers had continued work without respecting the Festivals or the general reality, which according to management was that

“a strategy must be ready before possible affected parties are involved. [if this had been respected] then the researchers would have avoided unrealis-

tic expectations [among the volunteers] and subsequent frustration and disappointment.” (Letter from the Festival management September 5, 1995, my translation)

Designing computer support for pre-production work

The checklist became the central point in the design process, despite all the technological visions the researchers tried to introduce to the users. The most important reason for that was that the Sound and Light group already had a very strong vision about a relational database. One member of the Sound and Light group had earlier experiences with computer support for festival pre-production. This support was implemented in a relational database management system, and looked very much like a “smart checklist”. The Sound and Light group had discussed this concept and made a sketch of a relational database screen layout as they would like it. This database sketch was basically a slightly expanded transformation of the checklist. The entire design project can be analysed as a transformation of the checklist (Bertelsen, in prep)

According to the original plan, design together with the users was to take place as a series of workshops; only one of these was realised. This Workshop took place in The Festival buildings, and was scheduled to cover 5 hours. The planned participants were members of four Festival operation groups: Sound and Light (3 persons), Yellow stage (one person), Catering (one person), and Transit (one person), plus the three seniors researchers and two apprentice participatory design researchers.

The plan for the workshop was to enact or simulate a series of work situations, both routine and problematic, from the planning (pre-production) and production of the festival. The participants were encouraged to bring real or made up scenarios that they found interesting, “focusing on the exchange of information” and how IT can be used, to the workshop. The idea was furthermore that the researchers would introduce various kinds of technologies into the game to elicit how, e.g. computerised telefax, central and local databases, e-mail, or hypermedia would change work at the Festival (Kyng 1995, Ehn & Sjögreen 1991).

The workshop was situated around a table with large pieces of paper mounted on the surrounding walls. One piece of paper was laid out with columns for various kinds of technologies; local databases, centralised databases, hypermedia, computer integrated telefax, etc. Cardboard lids were available to be used as database mock-ups, and yarn for simulating hyper-links between documents. Other pieces of wall paper were used to record situations and problems during the workshop. Material from the previous year’s Festival was photocopied in advance together with some made-up ideal typical material produced by the Sound and Light group.

The first problem the researchers encountered at the workshop was that the participant from the Yellow stage never showed, after an hour of waiting and several phone calls, his seat was filled out with one of the Sound and Light guys who had previously worked at the Orange stage group. This changed the balance of the workshop dramatically in a Sound and Light, and planning direction, and it became much harder to generate situations where the stage claimed not to have the information they needed. These situations would probably have arisen if the activist from Yellow stage had participated, because that group emphasised the lack of information during the preceding interviews.

The simulation games ended up focusing on how things were done the previous year; the workshop basically became a discussion repeating the information the researcher already got from the interviews. The cardboard lids and the yarn were never used, and the technology wall paper did not make its way into the situation. The design, or construction related part of the workshop was limited to the last half hour, when the original database sketch, produced by the Sound and Light group was examined with respect to suppliers and users of the information. This part of the workshop was important for building a prototype, but it did not break the meeting-ness of the workshop.

The design of the prototype took place right after the workshop. The first step was to make an object oriented description of pre-production and production, based on OMT (Rumbaugh et al. 1991). The main functions of this description became to generate discussions between the researchers about data formats, and to serve as a vehicle for the establishment of a shared understanding of The Festival among the researchers. In this process the understanding of the Festival the researchers got from the interviews was an important resource.

The transformation of the object-oriented description was done by mapping objects to tables in a straightforward manner. The issue of data-ownership distribution of the database over several non-networked PCs was already dealt with in the object-oriented model by reflecting the ownership of data in the division of objects. The construction of the user interface of the prototype started out on paper, but the researchers soon agreed that it was easier to program the interface directly without making a specification first. The task was uncomplicated because most of the prototype was specified in the Sound and Light database sketch, and on the pre-printed checklist made by Sound and Light the previous year.

The use of The DBMS yielded the possibility of designing the prototype interface directly on the computer without separate specifications; but at the same time, design was constricted by the lack of features for distribution in the database tool. This was obviously, at least seen in retrospect, a dangerous cocktail. To some degree, the technical limitations of the design artefact,

The DBMS, and not the obtained knowledge about The Festival determined the design. This unfortunate situation could have been avoided, had the researchers had a design artefact directly prescribing how to design user interfaces, as opposed to The DBMS, which prescribed this at a more implicit manner. In general, the absence of explicit theories and methods leaves room for the implicit world views of, e.g. the database tools to take over (see Bertelsen 1994, for a discussion of theories as design artefacts).

Contradictions and Activity systems

In the following sections the Festival project will be analysed in terms of contradictions. Contradictions are oppositions between things, concepts, persons, etc. E.g., good/bad, old/new, female/male, red/green, night/day desire/reality, labour/capital. Partial synonyms for contradiction are; tension, contrast, opposition, dichotomy, conflict, disconnection. Contradictions are basic in understanding the world: structuralist anthropology and linguistics understand the world in terms of binary oppositions (e.g. Lévi-Strauss); in dialectical thinking (Hegel, Marx etc.), the world in general, and its dynamics in particular is understood as the eternal resolving of inner antagonist contradictions. From a common sense perspective, a definition of something will usually specify what the defined is not, e.g. a community is always by the exclusion of others.

In the critical Scandinavian system development tradition (see e.g. Bansler 1987) the notion of hard contradictions and conflict has been predominant. Organisations have been understood as a battlefield, or frame of conflict (Borum and Enderud 1981), and ideal design as the negotiation between conflicting parties (Ehn & Sandberg 1979).

Engeström (1987) classifies contradictions in, and between activity systems. The basic unit of analysis in activity theory is human activity (e.g. work), i.e. the endeavour of a group of people to realise some object. Activity is socially mediated by artefacts, i.e. division of labour, rules, social formations, language and instruments. Engeström depict this unit of analysis as a triangular figure (fig 1), (For a thorough introduction to activity theory see e.g. Engeström (1987), Bannon & Kuutti (1990), Bødker & Bannon (1991), Nardi (1996)).

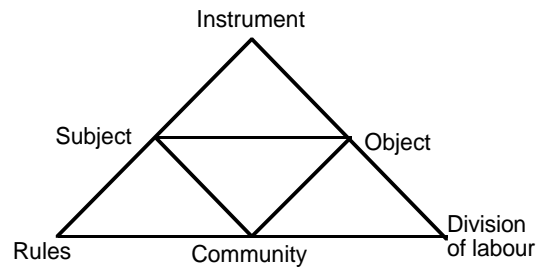


Figure 1: Leontjev's theory in Engeström's triangle.

Developmental research (Engeström 1987, 1990), deals with how contradictions, both internally in a considered central activity, and between the central activity and related activities, are the driving forces in development of the central activity.

The **primary contradiction** is the contradiction of commodity between use and exchange value. This double nature is a basic feature of the economic structure in capitalist culture; this contradiction penetrates every single corner of the triangle and is the basic source of instability and development (Marx 1962, pp. 49-55; Engeström, 1987 pp. 82-91).

The **secondary contradictions** of an activity system are contradictions between the corners of the triangle, e.g. between the skills of the subject and the instrument, or between rigid rules and new flexible instruments.

Tertiary contradictions are contradictions between the central activity and a culturally more advanced activity. Such contradictions can be generated by representatives from an other culture introducing culturally more advanced objects or motives into the central activity. The obvious problem is to know what is more advanced without subscribing to a very deterministic idea of history. However, the concept of a culturally more advanced activity does not necessarily imply historical determinism; it can also be interpreted as actual or potential different ways of conducting the central activity. Thus, the conflict between the history-determinism, still present somewhere in the basis of activity theory, and notions of heterogeneity (e.g. Star 1989) is not necessarily unsolvable.

Quaternary contradictions are contradictions between the central activity and the neighbouring activities; i.e.: 1) object-activities where the immediate objects of the central activity are embedded, 2) instrument producing activities where key instruments of the central activity are produced, 3) subject producing activities educating the subject of the central activity, and finally 4) rule producing activities like administration and legislation. An example of a quaternary contradiction is the contradiction between education of computer scientists at the university, focusing on mathematical formalisation and the central activity of computer scientists working as system developers in the industry.

An obvious (methodological) difficulty in applying activity theory to the analysis of a project like the Festival project is to identify one central activity.

Naturally the ‘neighbour activities’ also include central activities which are in some other way, for a longer or shorter period, connected or related to the given central activity, potentially hybridizing each other through their exchanges. (Engeström 1987, p.88)

Both the day-to-day work of the Festival, e.g. pre-production work, and the day-to-day work of design with users has to be considered. Furthermore the design artefact producing activity (what the researchers do at home) should also be included in the analysis. Instead of starting out by identifying a central activity, I start out by identifying contradictions in the project.

Contradictions in the project

In making sense of what happened during the Festival project the identification of contradictions became an important “tool”. This section describes contradictions in the Festival project identified without the use of a theoretical framework, and fit these contradiction into Engeström’s classification of contradiction.

Management vs. Volunteers

The initial impression of the Festival organisation was that it was totally harmonious; the motive shared throughout the organisation was to make a good festival; everybody had a say, no one gained economically from the Festival, most people were working voluntarily in the organisation, etc. Furthermore, since the initiative to start the project came from activists in the organisation, and later was approved by management, we believed that the project agreement reflected an objective shared by the whole organisation.

However, the most important contradiction in the Festival organisation, as it appeared during the project, is between management and the volunteers. Formally, the organisation has a very democratic structure; everybody have a say, the strategy of the organisation is set on annual meetings with all the operation groups etc. However, Festival management tries to maintain detailed control over the organisation and keep the level of information low on key issues. Management governs the organisation by decree and rigid rules, but due to management’s limited means of detailed control, the operation groups have a great deal of autonomy in how they carry out their tasks.

A lot of what is done by management and the booking group is known in detail only by these persons. Knowledge about which artists are being booked for the festival is the most prominent example; knowledge about which hotels the artists stay at during the festival is another. Obviously, there are good reasons for keeping this information secret. On the other hand, the organisation depends on the work and goodwill of the volunteers in the operation groups, who see themselves as necessary and trusted parts of the organisation. This **contradiction between openness and security** is an instance of the management versus volunteers contradiction. For example, a field for information about where the artists are accommodated, is present on the List of Acts (LoA), but this information is never filled into the LoA, distributed to the stage groups. The volunteers in the stage groups believed that this was due to technical difficulties. However, as the final lists, including hotel information, was typed by a secretary several days before the festival, this was not the real reason.

Another example of how the openness versus security contradiction is managed at the Festival by implicit and technical means, is that the volunteers are located in a building physically separated from management and the Festival secretariat. Thus technical limitations are used to cover up political decisions.

The contradictions between management and the operation groups are quaternary contradiction, management being the rule producing activity for the others. The confusing part is that the shared motive of these activities should be to make a good festival, however, this shared motive is primarily a part of the shared tales and anecdotes. The obvious main motive for some of the operation groups was to have a good time doing an interesting job together with good friends, whereas power internally in the organisation as well as externally, e.g. in relation to artist agencies, seemed to be the main motive of management. A more detailed analysis of such conflicting motives would generally be of great value as part of the establishment of a design project.

Use vs. Design

The basic contradiction in design is between use and design, both because design is a heteropraxial endeavour, involving designers from a design praxis, (the researchers), and users from the considered domain (the Sound and Light group); and because the temporary activity of design always will be opposed to the praxis which the object of design is aimed at supporting. This contradiction is a basic dynamic force in design, but it also has to be dissolved via simulations of work in design to ensure the suitability of the designed computer artefact.

The contradiction between use and design has a tertiary character because the contradiction impose learning and development on both sides. No one side of the use versus design contradiction can be regarded as culturally more advanced, but rather culturally different. However, it is also the matter of users having resources to invest in design activities; this was an obvious problem in introducing and evaluating the prototype in the Festival project.

In the Festival project, the tension between design and use never became dialectical because Festival Management prohibited the planned use of participatory design techniques.

Epaulettes vs. Utensils

The researchers understood the aim of the design project as one of developing a new tool which would make the planning and production of the Festival to proceed in a smoother and more efficient way; for them computers and computer based tools were useful tools for the organisation. However, this view was not shared throughout the organisation; especially the Festival management perceived computers as a reward to acknowledge volunteers who had done a good job in the organisation. This is illustrated through at least three examples during the project. The difficulties in having one of the purchased PCs allocated at pre-production office, were grounded in that the head of the Sound and Light group already had a paid job at the Festival, thus he did not need to have a PC also. Much the same problem was revealed during an interview with the general manager of equipment at the Festival; he had tried to convince Management of his need for a PC to keep track of a budget at more than 2 mill Danish kroners including e.g. 500,000 pieces of equipment, and beds for 500 persons. This was refused in spite of the fact that it in a longer run would result in decreased need for paid personnel. The clearest expression of Management's perception of computers as rewards appeared in their letter sent to the researchers at 5th of September. Here, Management states that because the researchers had involved parties possibly affected by the future IT-strategy, these parties had acquired expectations to get computers, and was thus likely to be disappointed or even frustrated.

Bjerknes & Bratteteig (1988) refers to the contradiction in systems development between what is technically interesting for system developers (or researchers?) versus what is applicable for users, as a contradiction between computers as epaulettes versus as utensils. To some degree, this contradiction also existed in the Festival project, and will always be present in a research project due to the **contradiction between the researchers' and the users' motives**. This is the reason why the Festival project can be both a failed design project and a successful research project.

I will adopt the notion of utensils versus epaulettes to describe the contradiction in the Festival project between computers as useful tools versus as rewards to the volunteers. This contradiction is an instance of the primary contradiction of commodity between exchange value and use value.

One important reason for the IT-initiatives in the Festival organisation was that the volunteers wanted computers as a new kind of rewards. However, this double nature of computers as being both epaulettes and utensils was not fully realised by the researchers during the project.

Good solutions vs. Easy solutions

The guiding principle in deciding which hardware and basic software configuration to buy, was convenience. Thus the Festival or the internal IT-group did not make any assessment of different configurations and architectures suitability for the actual situation in the Festival organisation, but went directly on to buy a configuration which was a copy of the configuration the head of the internal IT-group was managing in an engineering firm.

This is a kind of quaternary contradiction between the introducing-IT-in-the-Festival activity and the managing-IT-in-the-engineering-firm activity, the second producing the IT skills of the head of the Festivals internal IT group. This contradiction can also be seen as the contradiction between the central activity of the IT group and the object activity using IT in the Festival. Furthermore, it is a contradiction between the researchers Festival-project activity, regarded as central, and the activity of the IT-group as instrument producing.

Planning vs. Production

The cyclic nature of festival work was a source of unexpected problems in the project. Not only was it impossible, or at best difficult, for the researchers to learn about the different phases of work at the Festival before these actually happened. The project started when planning of the next festival was just about to start, at that time it was very hard to get any impression of how hectic things would be when the festival approached, not to mention during the festival days. But it was not only hard for the researchers to understand, the people at the Festival also belonged to different praxises during the year. A lot of anecdotes were shared in the operation groups and across operation groups, serving as secondary artefacts (Wartofsky 1973, see also Bertelsen in prep) maintaining skill and experience; these anecdotes were a very important source of knowledge for the researchers. Thus the operation groups were able to tell a lot about the Festival, but when it came to actual enactment of work during the workshop, it was nearly impossible for the Festival people to engage in situations other than the actual.

According to what the Sound and Light group told the researchers early in the planning period, their main task during the festival would be to drink coffee and listen to music, but according to what was actually observed during the festival, they were very busy solving all kinds of predictable and unpredictable problems.

This heterogeneity of festival work across the year was further emphasized during the workshop as none of the stage group activists were present, thus creating a very strong bias on planning versus production.

The contradiction between planning and production can be seen as quaternary, planning activities producing instruments for production activities, one of which have overlapping subjects.

However, in a Sound and Light group perspective both planning and production are directed towards the same motive, e.g. making a good festival, indicating that they are different sides of the same activity. Thus the Festival project points to a weakness of the activity triangle approach in analysing activities with a cyclic nature.

Design situations vs. Design artefacts

Due to the heterogeneity of the Festival year, the researchers' methods for enacting work-like situations did not work as expected, pointing to the need for new instruments for the design activity, new design artefacts. A more straightforward instance of contradictions between design situation and design artefact was the contradiction between the DBMS and the need in the situation to support cooperation in a non-networked architecture.

The design artefact versus design situation contradiction is an unavoidable secondary contradiction between instrument and object, but it was also a quaternary contradiction between the production of instruments for cooperative design in a non-cyclic setting, and the use of these artefacts in the Festival project.

In general, a contradiction between actual design praxis and prescription embedded in the design artefacts should be expected (Bertelsen, in progress).

Discussion

Systematic application of the notion of contradiction yielded important knowledge about the structure behind fundamental obstacles in the Festival

project. The activity theory framework served as a good way of structuring the analysis.

The primary contradiction between use and exchange value was present as a contradiction of computers as being both utensils and epaulettes. The only secondary contradiction identified in the analysis was between design and the existing methods for enactment of work-like situations. As the project aimed at developing new instruments for pre-production, secondary contradictions between the available instruments and other corners of the pre-production activity should be expected. However, being a computer scientist, the author did not see a need for new instruments as a necessary precondition for the initiation of a design project. A detailed analysis of contradictions between instruments and other corners of a considered activity will be a valuable resource in a design project. However, secondary contradictions emerged in the Festival activities, due to general knowledge among volunteers about how tedious office work can be done easier with computers. Thus tertiary contradiction subsequently induced secondary contradictions in Festival activities. The relation between use and design in the project has some elements of a tertiary contradiction, but in relation to the development of computer support in the Festival, the surrounding culture, e.g. at the volunteers daytime workplaces, was clearly examples of more advanced activities. Most of the identified contradictions were quaternary, which should come as no surprise since the analysis did not depart from the identifications of one central activity.

The main difficulties in applying the activity theory framework for analysis of the project are related to the heterogeneity of the considered project. When heterogeneity can be captured in terms of quaternary contradictions, the Finnish approach (Engeström 1987) is very fruitful, but if the considered activity systems have the cyclic nature of the festival activities, the Finnish approach is not straight forward to use. In terms of process-structure and structure-process diagrams (Mathiassen 1981), the problem is that the Finnish triangle models do not capture the situation when the considered activity is a structure subsuming a processual succession of more or less stable structures.

Thus key definitions in the Finnish approach may have to be adjusted or re-interpreted. Firstly, The notion of a more advance activity should be re-interpreted in a direction emphasising heterogeneity on the expense of historical materialism. Secondly, analysis of the Festival project showed the need for methods to deal with interwoven activity systems, and activity systems that have a very cyclic nature. It will be possible to deal with both these issues without rejecting the basis of activity theory, e.g. by incorporating Leigh Star's (1989) notion of boundary objects (see, Bertelsen in prep, for a discussion of boundary objects in the festival project). Thirdly, the Festival project showed that politics and power are issues which it is necessary to take into

consideration, but due to its strong emphasis on production, activity theory did not yield much in the analysis of this and other strange cultural phenomena.

Management's constant obstruction of the project was to some extent grounded in false fear of the consequences of new technology, thus some barriers could have been overcome by applying socio-technical techniques (e.g. Mumford & Ward 1968) for motivation and elimination of false fear. We may, humorously, call this a reversed socio-technical approach. However, contradictions between parties with different amount of power in the Festival organisation was the one most important reason why the project did not become a successful design project, despite the absence of traditionally recognised sources of conflicting interests, e.g. private economic gain. Thus, the project indicates that power struggle is a universal feature of organisations that should be made explicit and controlled by organising the design project as a negotiation between conflicting parties (Ehn & Sandberg 1979).

When developing computer support for praxes with a strongly cyclic nature, it is difficult to establish simulated work situations, because the different phases of the work cycle are only present during the full cycle in the form of anecdotes. Thus simulations of work are likely to degenerate into discussions about work. The project points to a need for the development of methods for simulation of work in such situations.

As system development researchers, we tend to believe that computer support is a mean for improving production: computers are utensils used to carry out a job. It is not new knowledge that computers are also epaulettes; they have been so on managing directors' desks for years. The new is that this aspect of computer support seems to have reached the floor. In the future, computer support will not only be a possible change of working condition and a de-skilling factor, but new computer technology will also be symbols of social status, epaulettes. In such a situation, the real challenge in system development may be to develop computer support without any functionality.

Acknowledgements

I want to thank the Sound and Light group and the other people from the Festival with whom we worked in the reported project; my co-workers in the Festival project, Morten Kyng, Preben Mogensen, Kim Halskov Madsen and Henrik Bærbak Christensen; Susanne Bødker, Jesper Just and the anonymous IRIS 19 reviewers for comments on a much earlier draft of this paper, Christina Nielsen and Niels-Oluf Bouvin for proof reading and comments on the present version; and finally, Jacob Bardram for comments and discussions.

References

- Bannon, L. & Bødker, S. (1991), Beyond the Interface: Encountering Artifacts in Use, in Carroll, John M. (ed.) *Designing Interaction*, Cambridge: Cambridge University Press.
- Bannon, L. J. & Kuutti, K. (1993). Searching for Unity among Diversity: Exploring the “Interface” Concept, In Ashlund, Stacey et al. (eds.), *INTERCHI '93: Conference on Human Factors in Computing Systems INTERACT '93 and CHI '93 Bridges Between Worlds*. New York, N.Y.: ACM Press.
- Bansler, J. (1989): Systems Development Research in Scandinavia: Three theoretical schools, *Scandinavian Journal of Information Systems* vol. 1, Aug. 1989, pp. 3-20
- Bertelsen, Olav (1994), Fitts' Law as a Design Artefact: A Paradigm Case of Theory in Software Design, in Blumenthal, Gornostaev, and Unger(eds.) *Human-Computer Interaction. 4th International Conference, EWHCI '94 St. Petersburg, Russia, August 1994. Selected Papers*, Berlin: Springer Verlag
- Bertelsen, Olav W. (in prep), The Festival Checklist: design as the transformation of artefacts. (not yet rejected conference paper)
- Bertelsen, Olav W. (in progress), Perspectives on design artefacts, ph.d.-thesis.
- Bjerknes, G. and Bratteteig, T. (1988): Computers—Utensils or Epaulets? The Application Perspective Revisited *AI and Society* vol. 2 no 3, pp. 258-266
- Borum, Finn and Enderud, Harald (1981), *Konflikter i organisationer: - belyst ved studier af edb-systemarbejde*, København: Nyt Nordisk Forlag
- Ehn, Pelle & Dan Sjögren (1991), From system description to script for action, in Greenbaum, Joan and Morten Kyng (eds.), *Design at work: cooperative design of computer systems*. Hillsdale: LEA.
- Ehn, Pelle & Åke Sandberg (1979) ‘God Utredning’ in Sandberg, Å. (ed.), *Utredning och förändring i förvaltningen*, Stockholm: Liber förlag
- Engeström, Yrjö (1987) *Learning by expanding: an activity -theoretical approach to developmental research*. Helsinki: Orienta-Konsultit Oy.
- Engeström, Yrjö (1990) *Learning Working and Imagining: twelve studies in activity theory*. Helsinki: Orienta-Konsultit Oy.
- Kyng, Morten (1994), Making Representations Work, in Suchman, Lucy (Ed.) (1995), Representations of Work, Guest edited section of *Communications of the ACM*, vol. 38, no 9, pp. 46-55.

- Leontjev, A. N. (1978). *Activity, consciousness, and personality*, Engelwood Cliffs NJ: Prentice Hall.
- Marx, Karl (1962) *Das Kapital*, vol. I, Berlin: Dietz Verlag
- Mathiassen, Lars (1981), *Systemudvikling og Systemudviklingsmetode* [in Danish: System development and system development method], Aarhus: Computer Science Department DAIMI PB-136
- Mumford, Enid & T.B. Ward (1968) *Computers: Planning for People* (quoted from the Danish translation, *EDB, system og menneske*, København: Steen Hasselbalchs Forlag, 1970)
- Nardi, B. (ed.) (1996). *Introduction to Context and Consciousness: Activity Theory and Human Computer Interaction*, Cambridge: MIT Press.
- Rumbaugh, James, M. Blaha, W. Premerlani, F. Eddy, & W. Lorenzen (1991), *Object-Oriented Modelling and Design*, Prentice-Hall International Editions, 1991.
- Star, Susan Leigh (1989), The Structure of Ill-Structured Solutions: Boundary Objects and Heterogeneous Distributed Problem Solving, in Gasser, Les and Michael N. Huhns *Distributed Artificial Intelligence, volume II*, London: Pitman Publishers, pp37-54
- Vygotsky, L.(1978). *Mind in society: The development of higher mental processes*, Cambridge, MA: Harvard University Press.
- Wartofsky, Marx W (1973), Perception, representation, and the forms of action: toward an historical epistemology, in Wartofsky, M. W., *Models*, Dordrecht: D. Reidel Publishing Company, 1979.
- Winograd, T. & Flores, F. (1986). *Understanding Computers and Cognition — a new foundation for design*. Norwood, N.J: Ablex.

Fifth paper

The Festival Checklist: design as the transformation of artefacts

Abstract: The idea that computer systems should be designed through abstract representations of the use domain has been systematically challenged in the Scandinavian tradition of user involvement in design. In this paper it is suggested that ideal design should happen as a transformation of artefacts rather than through abstract representations of domains. Thus the paper emphasises that design and design representations have to be tied to concrete material reality. The transformation idea is based on notions of secondary artefacts and boundary objects. The general ideas in the paper are introduced through a case story about a design project involving a music festival organisation and a group of university researchers, aiming at developing computer support for the planning and production of the festival.

Keywords: Transformation, representations, design artefacts, user involvement, secondary artefacts, boundary objects, case story.

Introduction

Traditionally, methods for development of computer artefacts have consisted of a, possibly iterative, sequence of the following steps. First, the considered domain of the future system is analysed, then abstract representations of the domain are constructed, which in turn are the basis for abstract representations of the system to be . Finally, the system is implemented by constructing material artefacts based on the abstract representation, and then introducing these artefacts into the considered domain. However, e.g. in the UTOPIA project (Bødker et al., 1987), it turned out that abstract repre-

sentations are very difficult for users to understand, thus there was a need for methods offering a more concrete way of investigating the future, past, and present. This paper aims at a further investigation of this theme by introducing the notion of transformation of artefacts as an alternative to the traditional emphasis on the representational aspects of design.

The first half of the paper reports on an action research project together with a music festival organisation. It tells a story about an artefact, the festival checklist, which emerged in a corner of an organisation as crystallised work praxis, and then made its way to other parts of the organisation. At a point the checklist made its way into a design project, in form of a database interface proposal. This proposal became the central document in the design of a new computer artefact which, among other things, was able to print out checklists. In the second half of the paper the transformation idea is developed based on activity theory (Engeström 1987), secondary artefacts (Wartofsky 1973), and boundary objects (Star 1989).

The Festival

The Festival is a non-profit organisation with the production of an annual music festival as its main objective. In 1995 the festival took place during 4 days, with concerts on 8 different stages, presenting a total of more than 140 different acts. Making a music festival involves many different tasks: engaging the artists, establishing camping areas for the audience, selling tickets, selling foods, controlling access to the festival site, informing the press, building the festival site, etc. The volunteers working in The Festival are organised in 35 operation groups, 150 persons are working throughout the year and during the festival additional 2500 volunteers are enrolled. 9000 members of external organisations (e.g. boy scouts and sports clubs) are working during the festival. 10 people have a regular, paid job at The Festival.

The focus of the project was on technical production and pre-production, involving people from the Booking group, Sound and Light group, Transit group, Catering group, and the eight stage groups (Green, Red, etc.). The Booking group is responsible for deciding which artists are going to play at the festival, and for negotiating the conditions and prices with the agents. The Sound and Light group is responsible for the technical side of the artists' performances. They are responsible for making arrangements with sub-contractors running the basic sound (PA) and light equipment on the individual stages, and for making sure that all the artists will have the conditions needed for their performances, equipment-wise, including instrument amplifiers, piano tuners, and help with special effects. In addition, the Sound and Light group has a co-ordinating role in the pre-production. The

Transit group is responsible for the transportation of the artists from airport to hotels to festival, etc., and the booking of hotel rooms. The Catering group is responsible for dressing rooms, meals, and other backstage facilities for the artists. The festival takes place at eight different stages. The stage groups are responsible for the production at the stages including establishing the facilities at the backstage area, e.g. stage and production offices, stage hands, etc.

The Project

The project took place during the first half of 1995, and involved a group of five researchers from a university, and various members of operation groups from a music Festival organisation. In the fall of 1994 The Festival decided that they needed an IT-strategy: an internal IT group was formed, and The Festival contacted the researchers to initiate a project eliciting the possible advantages of introducing IT in the planning and production of the festival. The project was initiated by the festival, because of a widespread expectation in the organisation to the possible benefits of the introduction of computer support. E.g. smoother coordination of work between different groups, and the possibility of getting rid of tedious manual routines.

The first meeting between The Festival and the researchers took place in the middle of January 1995, in the beginning of February the researchers decided to engage in the project, and an agreement for the project was formed. The first project meeting at The Festival took place at the end of February. The Sound and Light group told the researchers about The Festival from their perspective, and about their work. The researchers demonstrated some of their own software as inspiration for the Sound and Light activists. The Sound and Light group had prepared for the meeting, by making two descriptions on paper, one describing the “flow of information” to and from the Sound and Light group during pre-production, the other a sketch of a database for pre-production represented as a screen layout (figure 2).

During March a series of interviews with two of the stage groups, the Catering group, the group responsible for access to the festival area, the Transit group, the Booking group, festival management, and a secretary, were conducted. On the first of April a workshop with Sound and Light, Transit, Catering, and the Yellow stage groups took place. After the workshop the researchers decided to use a database management system (hereafter The DBMS) to have a prototype ready before the big rush of the pre-production activities. During April the researchers designed and implemented a first prototype of a system for Pre-production.

In the middle of April, the Festival management became nervous about the project, fearing that too much information would flow too freely around in

the organisation, therefore they dictated that the project could only continue with the Sound and Light group. As a consequence the second and third planned workshops with the operation groups had to be cancelled. This breach of the original agreement, made it difficult for the researchers to continue the work in a decent manner, especially it became impossible to confront their understanding of the festival with the actual reality.

During May, the first version of the prototype were installed at the Sound and Light office, and some of the existing data was entered into the system by the researchers. This version was never used by the Sound and Light group. At the end of May, a revised (simplified) version of the prototype more suited for the situation with Sound and Light as the only users was installed. At this time, Sound and Light had not made the checklists as they did it during the previous year's pre-production, some of the data were entered into the database, but most information were only available on faxes, and ad hoc notes. The final result was low quality checklists delivered to the stage groups.

During the last week of June, Festival 95 took place. The researchers conducted field studies at the Festival site. On the 25th of August the researchers sent the first version of a report on ways to improve the work of The Festival by the use of IT.

Pre-Production Work

During the spring, the head of the Sound and Light group is employed at The Festival to take care of the technical pre-production, and to distribute relevant information to other operation groups. The most important means of communication throughout this process is telefax, and to some extent telephone. Pre-production work is a kind of detective's work; when an artist is booked for the Festival and pre-production starts, the normal situation is that the only information the Sound and Light group has, is the name of an artist agent somewhere. Thus the first difficult task in pre-production is to find somebody who actually knows something about the artist and then to convince this person that The Festival needs up-to-date information as soon as possible. Pre-production work is complicated by several factors. People in the music business are always late with everything; it can be hard to make people understand that The Festival needs information in advance. Also, it is very important for especially the bigger artists to show off by demanding specific resources for their appearance at the Festival, these demands then have to be negotiated in some way. Finally, information about the Festival program, and information about arrival times and hotels of the artists has to be treated confidentially, both to maintain the advertising value of a coordinated release of program information to the public, and to protect the artists during the booking negotiations and during the festival. The general understanding in the Sound and Light group is that the festival could be

produced without pre-production, but that it then would be more chaotic. Thus, for the Sound and Light group, the purpose of pre-production is to facilitate a smooth production with a relaxed and friendly atmosphere.

**Check list
for artists at Festival**

Preproduction:

Name of artist: _____

Stage: _____

Date: _____

Time: _____

Tour/production manager: _____

Phone/fax: _____

Sound/light/monitor engineer: _____

Phone/fax: _____

Stage plan: _____ date _____ Sent to F-sound company: _____

Light plot: _____ date _____ Sent to F-light company: _____

Backline required: _____ Sent to Soundforce: _____ Confirmed: _____

Price: _____ We pay: _____ Artist pay: _____


Risers required: _____

Piano tuner: _____ Time: _____

Specials: _____

STAGE PRODUCTION: TURN PAGE.....

LOGO



STAGE PRODUCTION

Band: _____

Tour/production manager: _____

Get-in time: Crew: _____ Number: _____

Get-in time: Band: _____ Number: _____

Minutes on stage incl. encores: _____

Dressing rooms: Number: _____ Time: _____

“ “ “ “ “ “

“ “ “ “ “ “

Special agreements about photographers: _____

Special agreements about security: _____

Merchandise sale: Yes: _____ No: _____

If yes: agreement has to be made: _____

If yes: merchandise sold by: _____

Special agreement about TV/Radio: _____

List for KODA: _____

Transportation: _____

Flights: Arrival/departure: _____

Staying at hotel (name & phone): _____

Figure 1: The Sound and Light 94 checklist.

During pre-production Sound and Light builds a band file, a plastic folder enclosing documents, for each performance. These files are kept in a matrix of cardboard boxes, with one column per stage and one row per day. The first document in the file would normally be the checklist (see: below). One sheet of paper with the total plan of performances, the performance plan, organised in the same way is used both as a tool for locating the files in the cardboard boxes, and for recording central information about the specific performances, e.g. the state of the information gathering, and the need for special equipment. The performance plan is always situated on the desk in the Sound and Light office; when someone calls on the phone the Sound and Light person will look at the performance plan, locate the artist in question, and examine the state of the pre-production for this performance; then he will take the file in the cardboard box while continuing the discussion on the phone.

The checklist

The checklist is a sheet of paper with pre-printed fields for information related to a specific artist's performance at the festival (see figure 1). It was

originally invented by members of the Green stage group. This list contained fields for all the information that should be available or collected when an artist arrived at the backstage area at the Festival site. The checklist was filled in when the Sound and Light group handed information from the pre-production over to the stage groups, and it was later used when the artist arrived. Subsequently, it was adopted by the Sound and Light group, and used during the pre-production process as the central overview of the individual artists. From 1993 the Sound and Light group produced a common checklist for all the stages, and filled in the available information about the artists before carrying the complete files over to the stage groups. Thus the checklist had three functions: as a tool for the collection of information during pre-production, secondly as a medium for forwarding information from the Sound and Light group to the stage groups and, finally, as a tool at the stages when receiving the artists and carrying out the performances.

Constructing the computer artefact

The checklist became the central point in the design process. The most important reason for that was that the Sound and Light group already had a vision about a relational database that was far stronger than the technological visions about hyper-linking, etc. the researchers tried to introduce. The Sound and Light group's technological vision originated from a member who had experience with computer support from the pre-production of an other music festival. This support was implemented by means of a relational database management system, and looked very much like a "smart checklist". The Sound and Light group had discussed this concept and made a sketch of a relational database screen layout as they would like it (figure 2). This database sketch was basically a slightly expanded transformation of the paper based checklist.

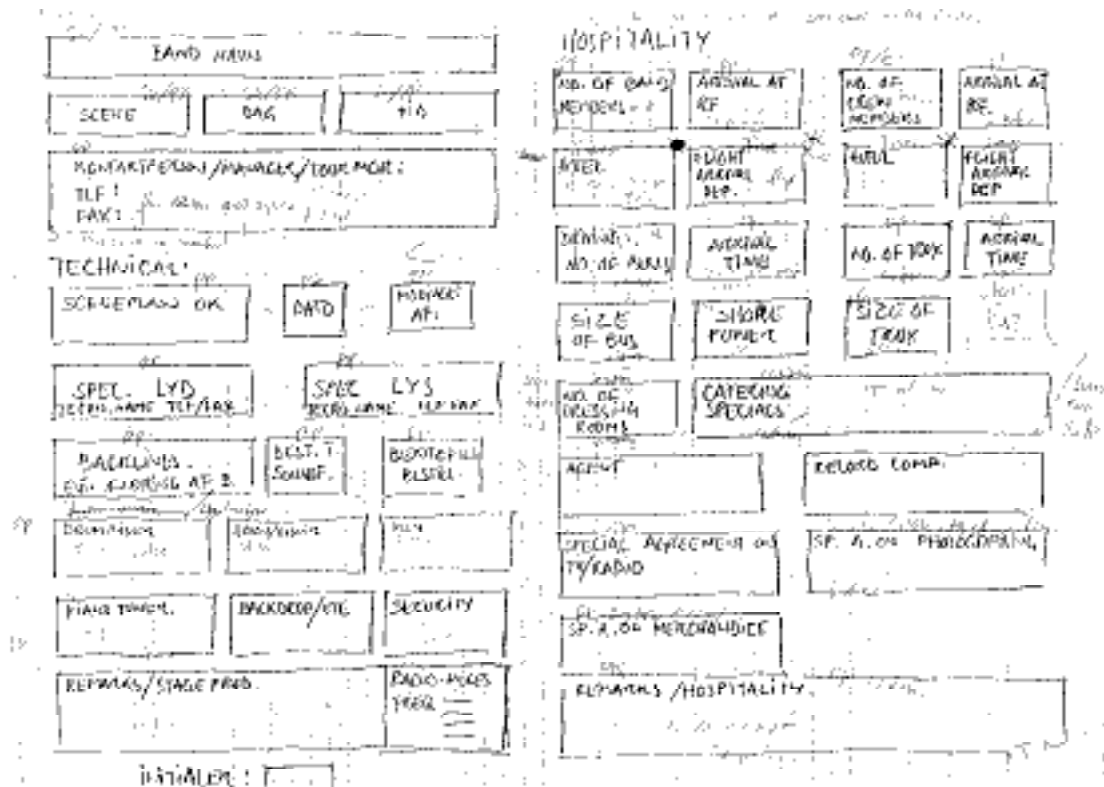


Figure 2: The database sketch.

The workshop

It was planned that design should take place together with the users in a series of workshops; unfortunately only one of these was realised. This Workshop took place in The Festival buildings, and was scheduled to 5 hours. The planned participants were members of four festival operation groups: Sound and Light (3 persons), Yellow stage (one person), Catering (one person), and Transit (one person); and five researchers.

The plan for the workshop was to enact or simulate a series of work situations, both routine and problematic, from the planning (pre-production) and production of the festival. The participants were encouraged to bring real or made up situations that they found interesting, "focusing on the exchange of information" and how IT can be used, to the workshop. The idea was furthermore that the researchers would introduce various kinds of technologies into the game to elicit how, e.g. computerised telefax, central and local databases, e-mail, or hypermedia would change work at the festival (Kyng 1995, Ehn & Sjögren 1991).

The workshop took place around a table, on the walls were mounted large pieces of paper. One piece of paper was laid out with columns for various kinds of technologies; local databases, centralised databases, hypermedia, computer integrated telefax, etc. Cardboard lids were available to be used

as database mock-ups, and yarn for simulating hyper-links between documents. Other pieces of wall paper were used to record situations and problems during the workshop. Material from the previous year's Festival was photocopied in advance together with some made-up ideal typical material produced by the Sound and Light group.

The first problem which the researchers encountered at the workshop was that the participant from the Yellow stage never came, after an hour of waiting and several phone calls his seat was filled out with one of the Sound and Light guys, who had previously worked at the Orange stage group. This changed the balance of the workshop dramatically in a Sound and Light, and planning direction, and it became much harder to generate situations where the stage claimed not to have the information they needed. These situations would probably have arisen if the activist from Yellow stage had participated, because that group emphasised the lack of information during the preceding interviews.

The simulation games ended up focusing on how things were done the previous year; the workshop basically became a discussion repeating the information the researcher already got from the interviews. The cardboard lids and the yarn were never used, and the technology wall paper did not make its way into the situation. The design, or construction related part of the workshop was limited to the last half hour, when the original, database sketch, produced by the Sound and Light group (figure 2) was examined with respect to suppliers and users of the information. This part of the workshop was important for building a prototype, but it did not break the meeting-ness of the workshop.

Building the prototype

The design of the prototype took place right after the workshop. The first step was to make an object oriented description of pre-production and production, based on OMT (Rumbaugh et al. 1991). The main functions of this description became to generate discussions between the researchers about data formats, and to serve as a vehicle for the establishment of a shared understanding of The Festival among the researchers. In this process the understanding of the Festival the researchers got from the interviews was an important resource.

The transformation of the object-oriented description was done by mapping objects to tables in a straightforward manner. The issue of data-ownership and access when the database was to be distributed over several non-networked PC's was already dealt with in the object-oriented model by reflecting the ownership of data in the structure of objects. The construction of the user interface of the prototype started out on paper but the researchers soon agreed that it was easier to program the interface right away without making a specification first. The task was, apart from data format issues, rela-

tively uncomplicated because most of the prototype was specified in the Sound and Light database sketch, and on the pre-printed checklist made by Sound and Light the previous year.

The use of The DBMS yielded the possibility of designing the prototype interface directly on the computer without separate specifications, furthermore the design was heavily influenced by the lack of features for distribution in the database tool. In retrospect, this was obviously a dangerous cocktail. The design artefact, and not the obtained knowledge about The Festival, determined design. This was both a result of technical limitations of the DBMS, and a result of the world view, and implicit prescriptions for design embedded in this design artefact. If the world view and prescriptions for design embedded in the design artefact had been more explicit, the conflict between this and the world view of the researchers would have been manifest, and then it would have been easier for the researchers to stick to chosen principles. This points to the general problem of implicit theories determining design (Bertelsen, 1994).

Using the computer generated checklist

As the Sound and Light group expected to get a working system, they did not use the printed paper checklist during the pre-production for Festival 95. Some of the pre-production information was entered into the system, but most of it was only present in the original letters, and telefaxes, and on the performance plan. Thus the Sound and Light group was in a dilemma at the time when they were about to hand the pre-production files over to the stage groups; should they abandon the design project and fill in paper checklists directly, or should they try to enter information into the prototype and print out the checklists. They ended up making the checklist via the prototype, which generated a lot of extra work because it was too late to use the information entered into the prototype for making lists and sum totals of, e.g. equipment requirements.

Possible reasons why the first prototype was never used by the Sound and Light group are that the facilities for getting the information entered into the system out on paper were not ready yet, and that the database was designed to support several groups' work with the pre-production information, thus the database was fragmented into various tables with their own screens, reflecting the ownership (right to update) of information, e.g. only Transit has the right to allocate hotel rooms.

The stage groups were disappointed with the checklists in the band files they got from Sound and Light prior to Festival 95. The 95 checklists did not contain as much information as the checklists from earlier years, but a lot of empty fields were also missing. Thus at least one of the stage groups made their own checklist in which they entered the information they got. This was

a surprise for the researchers, because they had seen the checklist exclusively from a pre-production perspective, thus overlooking that the checklist was also a *checklist* used when an artist arrived at the festival, i.e. the original use of the checklist.

This shows how unfortunate The Festival management's reduction of the project was. Had the project proceeded as planned with three workshops during the spring, the stage use of the checklists would have been elicited at a time when it was possible to change the design. A complicating aspect of the project with the festival was that festival work goes on in one-year cycles; thus versioning would take unrealistically long time. In such situations the use of simulated work situations is the only possible solution.

Transformation and Design

In the following sections the notion of design as a transformation process is introduced and exemplified by the Festival case.

Heteropraxiality and design artefacts

Design can be described as an activity where a designing subject designs the design object by means of some design artefacts. This activity is motivated by the artefact being designed, and it can be characterised as an aggregated meta-activity. However, this meta-activity does not exist as a concrete activity because the designing subject does not exist as concrete persons.

Design is basically heteropraxial, i.e., involving groups of people originating from different activity systems, (e.g. the researchers, the Sound and Light group, Festival management) in such a way that the individual activity systems can not be regarded as the basic unit of analysis. This heteropraxial nature of design is an obstacle in basing studies of design on approaches based on the identification of a central activity, e.g. Engeström's (1987) developmental work research. It is possible to get fruitful knowledge about what goes on in design by looking at the involved, often conflicting, activity systems, but it is difficult to identify a "central activity" with a uniform motive to base the study on. In the Festival project the checklist was created and recreated in a number of heterogeneous, and tightly intertwined activity systems, which were not simply ordered as central, instrument producing, consuming, etc.

To comprehend the central aspects of design it is necessary to apply a unit of analysis that transcends the division into activity systems. Thus I will suggest the perspective of the mediating artefacts in understanding design.

Design activity is mediated by design artefacts, utilised but not consumed during the process, serving as conditions or environment for the design process, thus opposed to materials. Examples of design artefacts are: object-oriented modelling techniques; principles of relational databases; The DBMS; material from last year's Festival; the concepts "situation", and "problematic situation"; technological visions (hypermedia, systems from other festivals, etc.); the semi-structured interview guide; a workshop layout; CSCW-perspectives, focus on communication and co-ordination, e.g. "shared material" (suggesting that the exchange of information between the operation groups was a bottleneck).

Design artefacts can be either general or local to a project. General design artefacts exist before the project, and are brought into, and utilised during the design process; generating a contradiction between what the design artefacts prescribe, and the praxis they induce (e.g. as a result of the contradiction between the central, and the instrument producing activities); and emphasising the role of basic assumptions (world views), and of explicit ideological statements. Local design artefacts are created inside a project, the database sketch is one example. Local design artefacts are often representations - descriptions or models - mediating the transformation of artefacts in the domain.

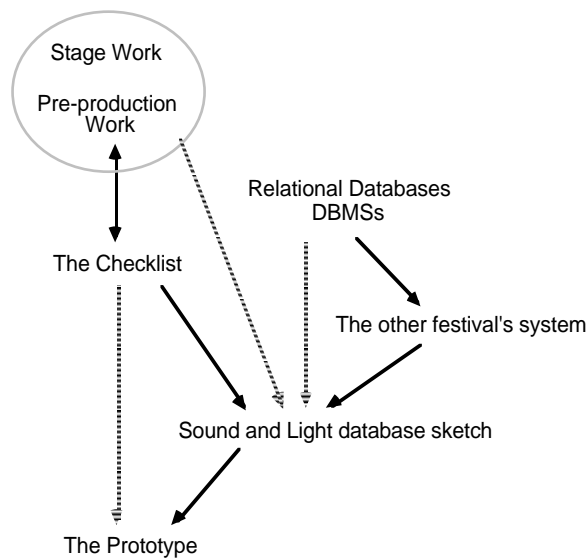


Figure 3: Influences in the development of the checklist. Solid arrows material transformation, dashed arrows indicate mediated relations.

Often the creation of local design artefacts is mediated by a general design artefact, e.g. the creation of the database sketch was mediated by general relational database concepts. In the festival case, the general relational database concepts were mediated by the actual system from the other festi-

val, in the creation of the database sketch. This kind of multiple mediation is illustrated in figure 3.

Design as transformation

Most system development methods put strong emphasis on representations, by basing design on descriptions of the domain of the future system (e.g. Jackson 1983, Mathiassen 1981, Rumbaugh 1991, Yourdon 1982). The fundamental difficulty in verifying the sanity of such representations before the system is implemented, was one of the driving forces in the early development of participatory design within, e.g. the UTOPIA project (see e.g. Ehn 1988). However, during the last years the issue of representations has got renewed attention among researchers in the field of user sensitive design of computer artefacts. (e.g. Suchman (ed.) 1995). The project with the festival can be analysed as a gradual transformation of the checklist. This will emphasise that representations are objects and that these objects are related to the considered praxis; that representations are material.

In general, artefacts are crystallisations of the use of an earlier artefact (Bærentsen 1989, Bannon and Bødker 1991). The original checklist as it was made by the Green stage group was a crystallisation of festival work. They were doing the same things every year when they received the files from pre-production, and when the artists arrived at the festival. They made notes on sheets of paper and these notes gradually became more standardised and, in the end, the pre-printed checklist was made. Thus knowledge about how to receive the artists and what to look out for was embedded in the checklist. The use of the files without checklists was crystallised into the new artefact, the pre-printed checklist.

When the Sound and Light group took over the checklist, it was transformed from a local artefact supporting work in the Green stage group, into a general artefact used across different groups in the festival organisation. In this “new” form the checklist served a broader range of functions, it became a planning tool and a media for information exchange. The checklist became a boundary object (Star 1989, see below).

In the design process, the checklist was first transformed into the database sketch by the Sound and Light group; the checklist became a local design artefact. For Sound and Light it was an incarnation of a technological vision, and it was a reminder of pre-production work, and of how this work supports the production at the stages. For the researchers the database sketch was first a too narrow technical vision, but later, during the last part of the workshop, it became the specification for the prototype. The object oriented descriptions made by the researchers were transformations of the checklist in the sense that they were a step in the definition of the relational tables in the prototype. For the researchers the prototype was the new checklist, with

which the Sound and Light group could do everything they previously did with the paper checklist. An important aspect of the paper based checklist was that it was handed over to, and used by the stage groups. The way this was done with the prototype was that the information in the database was printed out on paper and attached to the band file. In this way, the reincarnated checklist returned to the stage groups, but for them it was not a checklist anymore, because it had become a mere printout of the pre-production database.

The transformation of the checklist is illustrated in figure 4. The big arrow is the checklist transformed over time, the ovals are the main actors in the transformation. The horizontal arrows at the left side indicate important general design artefacts. The grey cross indicates where the transformation was broken.

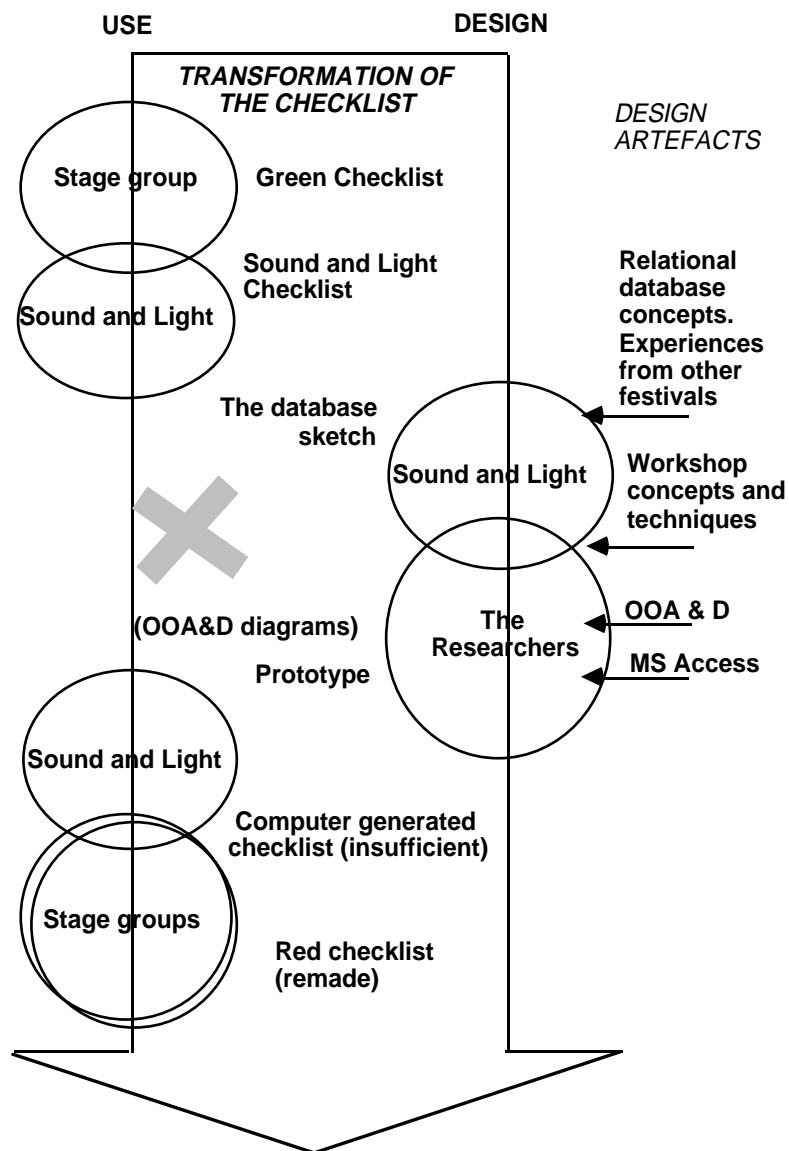


Figure 4: The checklist transformation history.

Representations: secondary artefacts and Boundary objects

In understanding design the role of representations—how they are related in general to human praxis—is central.

“...what constitutes a distinctively human form of action is the creation and use of artifacts, as tools, in the production of the means of existence and in the reproduction of the species. *Primary* artifacts are those directly used in this production; *secondary* artifacts are those used in the preservation and transmission of the acquired skills or modes of action or praxis by which this production is carried out. Secondary artifacts are therefore *representations* of such modes of action, and in this sense they are *mimetic*, not simply of the *objects* of an environment which are of interest or use in this production, but of those objects as they are acted upon, or of the modes of operation or action involving such objects.”
(Wartofsky 1973, p.202)

Representations and images are secondary artefacts, but secondary artefacts also exist in our heads related to our use of primary artefacts. Thus all artefacts have an element of secondary-ness. Design artefacts are both primary and secondary artefacts. As primary artefacts, design artefacts like CASE-tools and debuggers are very similar to hammers and spectacles. The specific features of design artefacts are tied to their representational function in design but, in general, design artefacts have a double role of being both primary and secondary artefacts. System descriptions are both specification of the new to be refined and filled out (primary artefact) and a place holder for knowledge and learning about the new (secondary artefact).

In the project, The DBMS served the double role as both mediating the production of machine executable code and as vehicle for the establishment of secondary artefacts, i.e. the relational tables as they were used for understanding the festival. In general, this double character is a problem in the design of computer artefacts, because the plasticity of secondary artefacts is obstructing or obstructed by the naturalism and formalisation of primary artefactness. In some situations, the formalised features are weak enough to allow the formalised description to acquire other meanings, in other situations, the models are so complicated that they are impossible to transcend because all attention is used in making sense of the formal contents of the figure. In some situations, formal descriptions offer openings into a poetic world of new possibilities, whereas in others they only offer the frustrating experiences of trying to understand the technical formalism the description is based on.

Star (1989) introduces the concept of boundary objects in trying to understand how people with completely different backgrounds working on different locations are actually able to work on the solution of the same scientific problem. This is very similar to the heteropraxiality of design work.

“Boundary objects are objects that are both plastic enough to adapt to local needs and constraints of the several parties employing them, yet robust enough to maintain a common identity across sites. They are weakly structured in common use, and become strongly structured in individual-site use. [...] a boundary object “sits in the middle” of a group of actors with divergent viewpoints. Crucially, however, there are different types of boundary objects depending on the characteristics of the heterogeneous information being joined to create them. The combination of different time horizons produces one kind of boundary objects; joining concrete and abstract representations of the same data produces another” (Star 1989, pp. 46-47).

Boundary objects depend on the plasticity of secondary artefacts related to the objects. But according to Wartofsky (op. cit.) representations are potential productive action. In the design of computer artefacts this relation between representation and action is more direct, or mechanical, because the representations in design can very often be turned into running programs in a formalised manner.

Design artefacts are boundary objects because they traverse the heterogeneous activity systems involved in the design process. When designers and users work together on a system specification, designers tend to perceive the specification as a sketch of the future program code, whereas the users may understand the specification in a less formalised, more open way.

Design artefacts do not only take different shapes or serve different purposes in different groups, they also change within one group, during the design process, and in the different rooms of a specific groups praxis. Thus a system development method is a boundary object in the sense that it has one function in the project organisation’s internal education prior to a project, and a totally different function during the project. Working in accordance with the method means two different things in the two project rooms. In a similar way, the checklist was a boundary object not only across different groups in the festival organisation, but also across the different stages of the transformation process.

In the project, the checklist existed both in the domain and in the design process (database sketch). It was a boundary object in the sense that it existed in different activity systems, and tied these together; across different activity groups; across the different, incommensurable stages of the individual groups cycle of the year; and across the transition from use to design. However, the checklist was not robust enough to be carried from the Festival, to the researchers and back into the Festival. In the transformation of the checklist from paper-based checklist into computer-based pre-production support it lost its checklist-ness. The researchers saw the checklist as a medium for the transfer of the information gathered together during pre-production, for them it was an incomplete, non-computer version of

the future computer system; but for the people working at the stages, the checklist had its main function as a tool for the preparation and reception of the individual artists. (Because of management's sabotage of the project this was not realised during the project)

The jungle is an artefact for the Natives living there, a crack of twig becomes the image of an animal to hunt and eat (Wartofsky op. cit). The jungle is also a boundary object; for the Native it is, among other things, a source of food; for the tourist it is an adventure. However, this boundary objectness breaks down when the paper industry represents the jungle as paper pulp. In the same way, the boundary objectness of the checklist broke down during the project, it lost the features that gave it meaning for the stage groups, the crystallised stage work was lost.

The researchers believed that the fields on the checklist, both paper- and computer-based, were place holders for the information filled in by Sound and Light, instead of understanding it in terms of the work crystallised into it.

The grey cross on the checklist transformation figure (figure 4) indicates that the design proposals were never confronted with actual festival work. The Sound and Light group was familiar with work in the stage groups, and the database sketch initially represented support for this work. However, without anchoring the representations attached to the database sketch all the relevant parts of festival work, they lost their boundary objectness.

For the transformation process to be successful, the artefacts must maintain their boundary objectness across both sites, and across design phases; and representations must remain representations of these artefacts or the activity crystallised into them. The cancelled workshops were intended to ensure this anchoring through the application of scenarios (e.g. Kyng 1995) and organisational games (Ehn & Sjögren 1991).

Discussion

Understanding design as transformation of artefacts is to emphasise material praxis, and that representations have to be understood in terms of the productive praxis they are aiming at. Thus, the transformation view becomes an ideal for the use of representations in design. In relation to object oriented methods, the transformation view rejects the idea that it is possible to base design on a general representation of the domain, e.g. a full description of The Festival (e.g. Coad & Yourdon 1990), and supports accounts focusing on modelling of artefacts like the checklist (e.g. Sørgaard 1988).

However, understanding design as transformation of artefacts also limits the innovative aspects of design. In future works, the innovative or creative side will be emphasised by introducing another class of design artefacts, generators as a complementary to the transformers dealt with in this paper. Examples of generators are future workshops (Jungk & Müllert 1987), and springboards (Engeström 1987). Generators will have an element of tertiary artefactness (Wartofsky op. cit.), i.e. mediate the creation of autonomous rooms for authentic creation, not related to the productive praxis in an obvious way.

The main shortcoming of the transformation perspective presented in this paper is that it does not incorporate an understanding of politics and power relations. In the project with the festival exactly such issues must be taken into account to fully understand why the transformation of the checklist broke down (Bertelsen 1996).

Acknowledgement

I want to thank the Sound and Light group and the other people from The Festival with whom we worked in the reported project; co-researchers in the Festival project, Morten Kyng, Preben Mogensen, Kim Halskov Madsen and Henrik Bærbak Christensen, and the group of “Systemarbejde 3” students at DAIMI who conducted interviews during the first phase of the project. Also thanks to Susanne Bødker, Jesper Just and the reviewers for comments on various drafts of this paper.

References

- Bannon, L. & Bødker, S. (1991), Beyond the Interface: Encountering Artifacts in Use, in Carroll, John M. (ed.) *Designing Interaction*, Cambridge: Cambridge University Press.
- Bertelsen, Olav (1994), Fitts' Law as a Design Artefact: A Paradigm Case of Theory in Software Design, in Blumenthal, Gornostaev, and Unger(eds.) *Human-Computer Interaction. 4th International Conference, EWHCI '94 St. Petersburg, Russia, August 1994. Selected Papers*, Berlin: Springer Verlag.
- Bertelsen, Olav W. (1996), Contradictions in the Festival Project - Activity systems, obstacles and dynamic forces in design, in Dahlbom, Ljungberg, Nuldén, Simon, Sørensen & Stage (eds.), *Proceedings of the 19th*

Information systems Research seminar In Scandinavia, (IRIS 19), 10-13 August 1996, at Lökeberg, Sweden.

Bærentsen, Klaus (1989), Menneske og maskine [Man and Machine], in Hedegaard, Hansen & Thyssen (eds.), *Et virksomt liv [An active life]*, pp. 142-187, Aarhus: Aarhus Universitets Forlag.

Bødker, Susanne, Pelle Ehn, John Kammersgaard, Morten Kyng, & Yngve Sundblad (1987), A UTOPIAN Experience: On Design of Powerful Computer-Based Tools for Skilled Graphic Workers, in Bjercknes, Gro, Pelle Ehn, & Morten Kyng (eds.), *Computers and Democracy*, Aldershot UK: Avebury

Coad, Peter & Edvard Yourdon (1990), *Object-Oriented Analysis*, Engelwood Cliffs N.Y.

Ehn, Pelle (1988), *Work-oriented Design of Computer Artifacts*, Falköping: Arbejdslivscentrum.

Ehn, Pelle and Dan Sjögren (1991), From system description to script for action, in Greenbaum, Joan and Morten Kyng (eds.), *Design at work: cooperative design of computer systems*. Hillsdale: LEA.

Engeström, Yrjö (1987). *Learning by expanding: an activity-theoretical approach to developmental research*. Helsinki: Orienta-Konsultit Oy.

Greenbaum, J. & Kyng, M. (eds.) (1991), *Design at Work*, Hillsdale, N.J.: LEA.

Jackson, Michael (1983), *System Development*, Engelwood Cliffs, N.J.: Prentice-Hall.

Jungk, R. & Müllert, N. (1987), *Future Workshops: How to create desirable futures*. London: Institute for Social Inventions.

Kyng, Morten (1994), Making Representations Work, in Suchman, Lucy (Ed.) (1995), Representations of Work, Guest edited section of *Communications of the ACM*, vol. 38, no 9, pp. 46-55.

Mathiassen, Lars (1981), *Systemudvikling og Systemudviklingsmetode* [in Danish: System development and system development method], Aarhus: Computer Science Department DAIMI PB-136.

Rumbaugh, James, M. Blaha, W. Premerlani, F. Eddy, & W. Lorenzen (1991), *Object-Oriented Modelling and Design*, Prentice-Hall International Editions.

Star, Susan Leigh (1989), The Structure of Ill-Structured Solutions: Boundary Objects and Heterogeneous Distributed Problem Solving, in Gasser, Les and Michael N. Huhns *Distributed Artificial Intelligence, volume II*, London: Pitman Publishers, pp. 37-54.

Suchman, Lucy (Ed.) (1995), Representations of Work, Guest edited section of *Communications of the ACM*, vol. 38, no. 9.

Sørgaard, Pål (1988), Object-Oriented Programming and Computerised Shared Material, in *Proceedings of ECOOP, Oslo Norway, August 1988*, Berlin: Springer Verlag.

Wartofsky, Marx W (1973), Perception, representation, and the forms of action: toward an historical epistemology, in Wartofsky, M. W., *Models*, Dordrecht: D. Reidel Publishing Company, 1979.

Yourdon, Edvard (1982), *Managing the system life cycle*, N.Y.: Yourdon Press.

Sixth paper

Organisational learning is crystallised into artefacts

Abstract: In this note I understand organisational learning in terms of manifest crystallisation of collective experience into artefacts, and I attempt to integrate an engineering and an emancipatory perspective. A Danish music festival serves as an example.

Introduction

Recently I attended a seminar on the Capability Maturity Model (Paulk et al. 1995), the current attempt to implement scientific management in the field of software development. The seminar took place in Copenhagen, requiring a four hours trip by train where I read a new paper by Yrjö Engeström (1996) given at this years congress on Piaget and Vygotsky in Geneva. The paper outlines a concept of development, radically different from how the concept was formulated in “Learning by Expanding” (Engeström 1987). Development is no longer depicted as a progression along the straight line of history; instead Engeström now understands development in terms of groups crossing borders in meeting different needs developing new motives.

Apart from being very stimulating for a general understanding of development, Engeströms Geneva paper also interfered with what I was going to hear about during the next couple of days. It appeared to me that both the CMM people and Engeström were addressing the issue of organisational learning.

CMM is a method for improving the working of software producing organisations; the goal is to be more rational, to be able to predict performance, to be able to avoid errors and re-work, to reduce cost, etc. An important element in the method is to measure what goes on and then *learn* from these measure-

ments. This can be seen as an engineering perspective on organisational learning; the organisation is a piece of machinery with definable objectives, and organisational learning becomes system optimisation. The obvious problem with the engineering perspective is that the direction and motivation of the organisation and of organisational learning is to be defined outside the organisation. Thus the engineering perspective do not yield any deeper understanding into the formation of directionality of development .

The emancipation perspective of Engeströms Geneva paper has the collective praxis of a group as it's main perspective, and expansive learning includes that the group transcends organisational and technical limitations. Organisational and historical structures are subsumed under the process of collective development of praxes. However, while this perspective offers an understanding of how motives and direction of development are formed, it does not deal with these issues in a way that is easily mapped onto the design of technical artefacts (including organisational structures).

Organisations are subsumed under the development of praxes, just as other technical artefacts are. Thus we can take organisational learning as a synonym for experience and development of a praxis made manifest in technical artefacts like company procedures, computer systems and standard document forms.

Organisational learning crystallised into artefacts

In trying to understand organisational learning in materialist terms, I emphasise that organisational learning is crystallised into artefacts. The remarks below are based on a recent project with a big Danish music festival (Bertelsen 1996a,1996b).

In the festival project, we developed a tool for cooperative planning of technical production at the music festival. This tool was based on an existing artefact, the checklist, that had emerged in the organisation as a crystallisation of what the organisation had “learned” about technical planning and production.

The checklist is a sheet of paper with pre-printed fields for information related to a specific artist's performance at the festival. It was originally invented by members of a group of volunteers working at one of the festival stages. This list contained fields for all the information that should be available or collected when an artist arrived backstage at the Festival site. The checklist was filled in when the people responsible for the technical planning of the festival handed the information over to the stage groups, and

it was later used when the artist arrived. Subsequently, it was adopted by the planning group, and used during planning as the central overview of the individual artists, and used as the central record when information was handed over to the stage groups.

The checklist had three functions: as a tool for the collection of information during planning, secondly as a medium for forwarding information from the group responsible for technical planning to the stage personnel and, finally, as a tool at the stages when receiving the artists and carrying out the performances.

In our project with people from the festival, the checklist took the form of a database interface proposal. This proposal became the central document in the design of a new computer artefact which, among other things, was able to print out checklists.

In general, artefacts are crystallisations of the use of an earlier artefact (Bærentsen 1989, Bannon and Bødker 1991). The original checklist as it was made by one of the stage groups was a crystallisation of festival work. They were doing the same things every year when they received the files from planning, and when the artists arrived at the festival. They made notes on sheets of paper and these notes gradually became more standardised and, in the end, the pre-printed checklist was made. Thus knowledge about how to receive the artists and what to look out for was embedded in the checklist. The use of the files without checklists was crystallised into the new artefact, the pre-printed checklist.

When the technical planning group took over the checklist, it was transformed from a local artefact supporting work in one specific stage group, into a general artefact used across different groups in the festival organisation. In this “new” form the checklist served a broader range of functions, it became a planning tool and a media for information exchange.

In the festival project we saw how learning in the organisation was crystallised into the checklist. Thus in taking organisational learning into account in the design of CSCW applications implies firstly that design is based on the transformation of the artefacts crystallising the organisational know how. Secondly CSCW applications should be able to enable continuous crystallisation of new modes of acting; thus pointing at the importance of issues like tailorability and code reuse.

Beside being crystallised into artefacts like the checklist the “lessons learned” in an organisation are crystallised into traditions. In the case of the music festival, traditions were the main “storage” for the organisations abilities. Although a lot of procedures were formalised into rules the basics of doing festival work was not made explicit. The festival organisation is very efficient in making the big annual festival. However, when the organisation for the first time engaged in making a one day festival with only one stage,

they had severe difficulties transferring this effectiveness. The problem in adapting to the new task of organising the smaller festival was that all procedures were hardwired into the organisation as unrecognised traditions. The festival organisation had been better able to handle the new task had the procedures been explicitly formalised. The problem of arranging the smaller festival was not in the lack of information, but in the failing ability to transfer existing procedures into appropriate action in the slightly changed setting. This suggests that an organisations ability to learn depends on a certain degree of formalisation and explicitness of modus operandi, parallel to the need for a developed orienting basis in the case of development of mental acts in individuals (Gal'perin 1969).

Conclusion

A concept of organisational learning informing the design of CSCW applications, has to reflect both engineering and emancipation. The notion of crystallisation of organisational learning into artefacts is suitable because it focuses on technical artefacts while taking praxis as the basis of analysis. Thus, avoiding to understand organisational learning as mere system optimisation or gathering of information, but rather as the active acquisition and construction of ability to act.

In reflecting a materialist understanding of organisational learning into design of CSCW applications we should maintain the procedural memory of the organisation by transforming existing artefacts crystallising this memory into new artefacts. Further we should support continual crystallisation of organisational procedures and knowledge into the CSCW application by facilitating tailoring and easy redesign. Finally we should support continued organisational learning by formalising and making explicit the current modus operandi by the CSCW applications. Thus making the CSCW application a Marxo-Freudian tool for building consciousness about the existing limitations in order to transcend these and build a new emancipated praxis.

References

Bannon, L. & Bødker, S. (1991), Beyond the Interface: Encountering Artifacts in Use, in Carroll, John M. (ed.) *Designing Interaction*, Cambridge: Cambridge University Press.

Bertelsen, Olav W. (1996a), Contradictions in the Festival Project - Activity systems, obstacles and dynamic forces in design, in Dahlbom, Ljungberg, Nuldén, Simon, Sørensen & Stage (eds.), *Proceedings of the 19th Information systems Research seminar In Scandinavia, (IRIS 19), 10-13 August 1996, at Lökeberg, Sweden.*

Bertelsen, Olav W. (1996b), The Festival Checklist: design as the transformation of artefacts. in *Proceedings of the Fourth Biennial Conference on Participatory Design Cambridge ,MA, USA 13-15- November 1996*

Bærentsen, Klaus (1989), Menneske og maskine [Man and Machine], in Hedegaard, Hansen & Thyssen (eds.), *Et virksomt liv [An active life]*, pp. 142-187, Aarhus: Aarhus University Press.

Engeström, Yrjö (1987), *Learning by expanding: an activity -theoretical approach to developmental research.* Helsinki: Orienta-Konsultit Oy.

Engeström, Yrjö (1996), Development as breaking away and opening up: a challenge to Vygotsky and Piaget, *in IInd Conference for Sociocultural Research, Geneva Switzerland September 11-16, 1996*

Gal'perin, P. Y (1969), Stages in the Development of Mental Acts, In Cole & Maltzman (eds.) *A Handbook of Contemporary Soviet Psychology*, New York: Basic Books.

Polk, M. C., Weber, C. V., Curtis, B., Chrissis, M. B. (1995), *The Capability Maturity Model: Guidelines for Improving the Software process*, Reading MA, Addison-Wesley.

Seventh paper

Understanding objects in use-oriented design

Abstract: The paper is an investigation into the relation between use-oriented design and object-oriented methods. Based on the notions of historical crystallisation of praxis into artefacts, the concept of secondary artefacts, and the notion of boundary objects, it is argued that object-oriented descriptions should be understood as transformations of artefacts from the considered praxis rather than models of the domain. The transformation idea is illustrated by a case story, from a design project with a music festival.

Introduction

A persistent trend in computer programming languages and software development methods is object-orientation. One of the basic concepts in object-orientation is physical modelling, i.e. structuring computer programs, and describing the real world, by constructing models as compositions of objects with physical properties.

From the very beginning physical models have been regarded as modelling something—the referent system—which according to (Holbæk-Hansen et al., 1975) can be either mental or manifest:

“Systems existing in the human mind, physically materialised as states of the cells of our brains, are called *mental systems*. Systems external to human minds are called *manifest systems*.” (ibid. , p.18).

A problem for the further advance of physical modelling based methods in the design of computer artefacts, is the absence of a consistent framework for understanding the relation between technical design and development of use praxis that transcends the mechanical idea of the referent system.

The question I try to answer in this paper is whether object-oriented methods are specially suited for involving users in design or not. At first glance the two seem to contradict due to the radical difference in the way the world is perceived according to object-oriented concepts and how it is understood by non-computer professionals accommodated in the real world. I will show that it is possible to establish an activity theory based framework for understanding the benefits of object-oriented methods in user oriented design of computer artefacts. The basic idea is to focus on the transformation of artefacts from the considered praxis, instead of focusing on the construction of abstract, although in concrete terms, models of idealised domains. Furthermore, it is argued that the principle of physical modelling is likely to support some aspects of cocreation (users and designers) of new technological visions.

The argument is based on three theoretical statements:

- Firstly, that activity with one generation of artefacts is crystallised into the next generation of artefacts (Bærentsen, 1989).
- Secondly, that artefacts mediating productive praxis also have a representational side, secondary artefactness, maintaining and transferring the modes of acting with the artefact, thus making a tight connection between perception and action (Wartofsky, 1973).
- Thirdly, that the coexistence of heterogeneous praxes is mediated through artefacts, boundary objects, that while stable across sites still have different meanings to the involved praxes (Star, 1989).

Object-orientation

There are many different definitions of object-orientation. However, in this paper I will stick to definitions of “object-oriented” in line with that given by (Madsen et al., 1993). According to Madsen et al., object-oriented programming means that:

“A program execution is regarded as a physical model, simulating the behaviour of either a real or an imaginary part of the real world”
(Madsen et al., 1993, p. 16).

This implies that programs are structured into objects that interact with each other only through well-defined interface to other objects and the surrounding world, and that the internal structure of the program can be perceived as modelling something. In some respects object-orientation has much in common with LEGO bricks.

Furthermore, object-oriented programming is based on concept building through classification and composition. Classification means that concrete phenomena are understood as instances of concepts, which in turn may be specialisations of more general concepts. E.g. Olav's bicycle is an instance of the class bicycle and the concept (or class) bicycle is a specialisation of the more general concept vehicle. Composition means that concepts and phenomena are aggregates of other concepts or phenomena. Thus a bicycle is an aggregate of wheels, frame, handlebars, pedals, transmission, etc.

Object-orientation has three benefits, still according to Madsen et al. (1993) which are:

- real-world appreciation,
- stability of design, and
- reusability.

This paper focuses on the claims about “real-world appreciation” since that is what most directly relates to the possible advantages of object-orientation seen from a user involvement perspective. Clearly re-use and design stability are also important issues for user involvement in the obvious sense that fast and safe is always a good thing. It is this apparent similarity with the real world that makes it promising to apply object-orientation in, not only programming, but also in earlier stages of system development: description of problem domain, often referred to as analysis, and description of the coming software, often called design.

Real world appreciation in object-orientation has two elements:

- natural concept building, and
- physical modelling.

Coad and Yourdon (1990) describe the real world appreciation of object-orientation in the following way:

“Object-oriented analysis is based upon concepts that we first learned in kindergarten: objects and attributes, classes and members, whole and parts.” (ibid., p.1)

“... OOA [object-oriented analysis] organizes analysis and specification using methods of organization which pervade people's thinking” (ibid., p.3)

The first quote explains why we may hope that users understand object-oriented models. However, the second quote is more problematic, because it implies that people perceive the world they live in by means of concepts with well-defined features. An object-oriented model of a bank account will consist of well-defined attributes such as owner, balance, etc., whereas several different definitions of a the concept account exist among the different

groups working in a concrete bank (Experience from a project with a bank, reported by Heinz Züllighoven and co-workers, at a research seminar at Århus University). Thus it can not be assumed that object-oriented models are complete reflections of concepts in the considered praxis. In praxis concepts have often no definition at all but work well anyway. In general, our day-to-day life is not governed by strict definitions and pure rationality; well-defined rationality is just a convenient artefact in some situations.

While the idea that concepts are well defined is not generally valid, the notion of concept building makes sense as an integrated part of physical modelling. Thus, in a physical model things are well defined in the sense that a model is a composition of objects that have well-defined properties, and interact in well-defined ways.

“Object models can represent the real world as you understand it. Specifically, the real world can be represented as a collection of interacting entities, each of which exhibits well-defined behaviours. It is not necessarily true that the world is a collection of interacting entities, but the object-oriented approach assumes that the aspects of the world in which you are interested can be modelled in this way. [footnote: This statement says that objects are not *in* the world but *of* the world]” (Goldberg & Rubin, 1995, p.50).

Physical models are not only used in the design of computer artefacts, they are an important part of most manual systems. Madsen et al. (Madsen et al. , 1993) use a traditional, paper based Norwegian train reservation system, and medical records as examples in introducing the idea of physical modelling. In the train reservation system the physical model is a representation of the train sets with wagons with seats that can be reserved. Similarly with the hospital record:

“... a medical record corresponding to each patient keeps track of the relevant information related to that patient. This record may be considered to be a representation of the patient.” (Madsen et al., 1993, p.16)

However, it must be noticed that representations are always constructed with a certain purpose in mind, and that they are part of a certain praxis (Wartofsky, 1973, see below). Thus, the medical record does not comprise all possible information about the patient but the information needed for the people treating the patient, thus the “representation of the patient” is a prescription for action on the patient.

Computerised physical models (or object-oriented programs) are composed of objects which can be considered computerised material. Objects have attributes of properties characterising the phenomenon being modelled, and they have certain well-defined ways of interacting with other objects and changing states of attributes. A computerised physical model can represent real world phenomena in the same way as the manual Norwegian train

reservation system or the medical record, but it can also model phenomena not existing in the real world.

Thus, physical models, computerised or not, are not disinterested, naturalistic description of a domain that can be used for what ever future action in, or change of the modelled domain.

Design of computer artefacts

Design is about creating conditions and environments for human life. In the design of computer artefacts we are mostly concerned with environments for work. Designing and introducing a new computer artefact into a given praxis is changing this praxis; development of technology is also organisational development.

One of the main issues in designing computer artefacts, is to ensure that the new artefact makes sense in the domain of future use. In the Participatory design tradition (Bjerknes et al. (eds.), 1987, Greenbaum & Kyng (eds.), 1991) this is done by involving users in the design process and letting them codetermine what are suitable solutions.

The involvement of users in design can be motivated by a general democratic concern; but more generally imperative, users have to be involved because it is impossible to formalise human activity to an extend where outsiders are able to verify the sanity of design proposals.

The theory of “tacit knowledge” (Polanyi, 1966) describes how it is impossible for members of a given praxis to explicate, not to say formalise, their own praxis. Thus, it is necessary to use non-formal and “hands on” experiences to ensure that the evolving artefacts make sense in the domain of future use. The impossibility of using formal system descriptions together with users was, e.g. recognised in the UTOPIA project (Bødker et al., 1987, Ehn & Sjögren, 1991).

The number of existing “methods” for object-oriented analysis and design is exorbitant. The classics include JSD (Jackson, 1983), that was not called object-oriented although it is in most respects; and Coad & Yourdon’s OOA (Coad & Yourdon, 1990). Among the newer methods OMT (Rumbaugh et al., 1991), Booch (1994), and the “use-case driven approach” (Jacobson et al., 1992) have acquired a leading position. However, most methods are primarily concerned with technical aspects, and are mostly differentiated by supporting specific programming languages, or by variations of diagramming syntax. Thus, most of the methods can be seen more as methods for “very high level programming” than as methods for design of environments and

instruments for human work. The recent standard proposal for a “unified method” is no exception from this.

Work activity and design activity

Design of computer artefacts is a process of changing a considered praxis through the construction and introduction of new artefacts, and the process of design is itself human activity. To understand the role which object-oriented methods and physical modelling play in this game the general notion of human activity is introduced below.

The basic unit of analysis in activity theory is human activity (e.g. work), i.e. the endeavour of a group of people to realise some object (here the meaning of “thing”). Activity is socially mediated by artefacts, i.e. division of labour, rules, social formations, language and instruments. (For a thorough introduction to activity theory see e.g. Engeström, 1987, Bannon & Kuutti, 1993, Bannon & Bødker, 1991).

According to Vygotsky (1978) Human activity has three fundamental characteristics;

- firstly, it is directed towards a material or ideal *object* which distinguishes one activity from another;
- secondly, it is *mediated* by artefacts (tools, language etc.); and
- thirdly, it is social within a *culture*.

Computer artefacts, like all other artefacts, in this way mediate human activity within a practice.

Human beings meet the objective world through acts in the world. Thus, human knowledge and experience about the world is *reflection* obtained through activity, forming the basis for expectations, and desires about possible activities in this world.

Human activity can be analysed as a three level hierarchy: *activities* realised through chains of *actions*, which are carried out through *operations*. At each of these levels the objective world is reflected through the activity. Human activity is always directed toward a material or ideal object satisfying a need. The subject’s reflection of, and expectation to, this object characterises the *motive* of the activity. Human activity is carried out through actions, realising objective *results*. These actions are controlled by the subject’s conscious *goals*. Goals are the reflection of the objective results of the action. Actions are realised through a series of operations; each determined by the concrete physical *conditions* of the action. These operations are performed

without thinking consciously but are oriented in the world by a non-conscious *orienting basis* of the operation. This orienting basis is established through experience with the concrete material conditions for the operation, and is a system of expectations about the execution of each operation controlling the operation, in the process of the activity (Leontjev, 1978).

Heteropraxiality of design

Design can be described as an activity, motivated by the artefact being designed, where a (collective) designing subject designs the design object by means of some design artefacts. However, design is basically heteropraxial, i.e., involving groups of people originating from different activity systems, in such a way that the individual activity systems can not be regarded as the basic unit of analysis.

To comprehend the central aspects of design it is necessary to apply a unit of analysis that applies across the distinct praxises involved. Thus I will suggest the perspective of the mediating artefacts in understanding design. Design activity is mediated by design artefacts, utilised but not consumed during the process, serving as conditions or environment for the design process, thus opposed to materials. Examples of design artefacts are: object-oriented modelling techniques; principles of relational databases.

Design artefacts can be either general or local to a project. General design artefacts exist before the project, and are brought into, and utilised during the design process; generating a contradiction between what the design artefacts prescribe, and the praxis they induce (e.g. as a result of the contradiction between the central, and the instrument producing activities); and emphasising the role of basic assumptions (world views), and of explicit ideological statements. Local design artefacts are created inside a project, they are often representations - descriptions or models - mediating the transformation of artefacts in the domain.

Crystallisation

In general, praxis is historically crystallised into artefacts, thus the historical development of praxis can be read from the development of artefacts mediating the praxis (Bærentsen, 1989, Bannon & Bødker, 1991).

“Artifacts can be characterised as *crystallized knowledge*, which means that operations that are developed in the use of one generation of technology are later incorporated into the artefact itself in the next”
(Bannon & Bødker, 1991, p.243).

Activity is crystallised into artefacts in two ways. Firstly as externalisation of operations with earlier artefacts, and secondly as secondary artefacts rep-

representing modes of acting in the given activity (see below). Artefacts mediating human activity are not just more or less suitable attachments to human praxis, but they are constituting activity. This suggests that the mediating artefacts could be regarded more stable than the object of the activity; this has consequences for the way physical modelling based techniques should be used in the design of computer artefacts.

Perception and action: secondary artefacts

In understanding object-oriented methods the role of representations in general human praxis is central.

“...what constitutes a distinctively human form of action is the creation and use of artifacts, as tools, in the production of the means of existence and in the reproduction of the species. Primary artifacts are those directly used in this production; secondary artifacts are those used in the preservation and transmission of the acquired skills or modes of action or praxis by which this production is carried out. Secondary artifacts are therefore representations of such modes of action, and in this sense they are mimetic, not simply of the objects of an environment which are of interest or use in this production, but of those objects as they are acted upon, or of the modes of operation or action involving such objects”
(Wartofsky, 1973, p.202) .

Representations and images are secondary artefacts, but secondary artefacts also exist in the praxis of using primary artefacts. Thus, all artefacts have an element of secondary-ness. Design artefacts are, in a more radical sense, both primary and secondary artefacts. They are primary artefacts because they mediate the productive act of constructing machine executable code, and they are secondary artefacts because they reflect modes of acting in the considered praxis as well as in design. An object-oriented model of a train will both be a primary artefact mediating the production of new software, and a secondary artefact preserving and transferring knowledge about, e.g. train booking, and mediating the construction of visions of a new artefact.

The Norwegian train and the medical record are examples of both primary and secondary artefacts. They are secondary artefacts because they prescribe acts in the world rather than merely describing it.

The double role of design artefacts as both mediating the production of machine executable code, and the establishment of secondary artefacts, e.g. relational tables from a CASE tool may be used for understanding the considered praxis and establishing visions of the new is a general problem in the design of computer artefacts; because the plasticity of secondary artefacts is obstructing or obstructed by the naturalism and formalisation of primary artefactness. In some situations, the formalised features are weak enough to

allow the formalised description to acquire other meanings, in other situations, the models are so complicated that they are impossible to transcend because all attention is used in making sense of the formal contents of the figure. In some situations, formal descriptions offer openings into a poetic world of new possibilities, whereas in others they only offer the frustrating experiences of trying to understand the technical formalism the description is based on.

Boundary objects

Design is complicated by the heterogeneity of the various groups participating. Star (1989) introduces the concept of boundary objects in trying to understand how people with completely different backgrounds working on different locations are able to contribute to, e.g. the solution of the same scientific problem. The heteropraxiality of design is similar to this.

“Boundary objects are objects that are both plastic enough to adapt to local needs and constraints of the several parties employing them, yet robust enough to maintain a common identity across sites. They are weakly structured in common use, and become strongly structured in individual-site use. [...] a boundary object “sits in the middle” of a group of actors with divergent viewpoints. Crucially, however, there are different types of boundary objects depending on the characteristics of the heterogeneous information being joined to create them. The combination of different time horizons produces one kind of boundary objects; joining concrete and abstract representations of the same data produces another” (Star, pp. 46-47).

Boundary objects depend on the plasticity of secondary artefacts related to the objects. But according to Wartofsky (1973) representations are potential productive action. In the design of computer artefacts this relation between representation and action is more direct, or mechanical, because the representations in design can very often be turned into running programs in a formalised manner.

Design artefacts are always boundary objects. They traverse the heterogeneous activity systems involved in design, taking differing meaning and function in each activity system: furthermore design artefacts are boundary objects due to their double character as both primary and secondary artefacts, both serving as “hammers” for constructing running computer programs, and “binoculars” for creating the future. When designers and users work together on a system specification, designers tend to perceive the specification as a sketch of the future program code, whereas the users may understand the specification in a less formalised, more open way.

Design artefacts do not only take different shapes or serve different purposes in different groups, they also change within one group, during the de-

sign process, and in the different rooms of a specific group's praxis. Thus a system development method is a boundary object in the sense that it has one function in the project organisation's internal education prior to a project, and a totally different function during the project. Working in accordance with the method means two different things in the two project rooms.

The jungle is an artefact for the Natives living there, a crack of twig becomes the image of an animal to hunt and eat (Wartofsky, 1973). The jungle is also a boundary object; for the Native it is, among other things, a source of food; for the tourist it is an adventure. However, this boundary objectness breaks down when the paper industry represents the jungle as paper pulp. In the same way, the boundary objectness of the artefacts transformed during the design process can break down by losing the features giving them meaning for the users, thus losing the work praxis originally crystallised into them.

Shared understanding, i.e. that designers and users establish a common understanding of the thing being designed, may not be important. Thus, instead of defining participatory design as a process of shared sense making, it is important to acknowledge that the "shared understanding" constructed during, e.g. a rapid prototyping session (Bødker & Grønbæk, 1996), in general only exists in a form crystallised into the prototype; the prototype in turn being a boundary object tying the incommensurable praxes of designers and users together, allowing them to design together but still perceive the situation and the new artefact in different ways. In this sense, the notion of boundary objects supports and explains the concerns stated by Ehn and Kyng (1987) about codetermination, mutual learning, etc. The whole point in doing design together with users is that designers will never learn to think like users, however they are able to engage in a fruitful cooperation.

Modelling as Transformation

Most system development methods put strong emphasis on representations, by basing design on descriptions of the domain of the future system (e.g. Booch, 1994, Jackson, 1983, Jacobson et al., 1992, Mathiassen, 1981, Rumbaugh et al., 1991, Yourdon, 1982). However, the fundamental difficulty in verifying the sanity of such representations before the system is implemented, was one of the driving forces in the development of methods for involving users in design. Computer artefacts can not be designed at the modelling board due to the fact that use aspects are constituted through the introduction of the artefact into cultural praxis. Thus the notion of transformation is specially important in the design of computer artefacts as opposed to building houses. For the transformation process to be successful, the artefacts must maintain their boundary objectness across both sites,

and across design phases; and representations must remain representations of these artefacts or the activity crystallised into them.

Understanding design as a transformation of artefacts from the application domain emphasises that representations are things and that these things are related to the considered praxis; that representations are material. Representations have to be understood in terms of the productive praxis they are aiming at; thus, the transformation view becomes an ideal for the use of representations in design.

Object-oriented, or physical modelling based, design artefacts can secure the sanity of the relation between design and domain, by allowing all involved parties to perceive, use and recreate the models in their own professional terms not fully understandable for the other involved groups. The physical model may change during the process, but it will maintain identity across participating groups at any given point in time. Physical models are in this sense instances that, due to their boundary objectness, mediate cooperation between parties without the same language.

The festival checklist

The role of crystallisation and transformation can be illustrated with a recent project with a big Danish music festival as an example (Bertelsen, 1996a&b, Madsen 1996).

In the festival project, we developed a tool for cooperative planning of technical production at the music festival. This tool was based on an existing artefact, the checklist, that had emerged in the organisation as a crystallisation of knowledge about technical planning and production.

The checklist is a sheet of paper with pre-printed fields for information related to a specific artist's performance at the festival. It was originally invented by members of a group of volunteers working at one of the festival stages. This list contained fields for all the information that should be available or collected when an artist arrived backstage at the Festival site. The original checklist was filled in when the people responsible for the technical planning of the festival handed the information over to the stage groups, and it was later used when the artist arrived. Subsequently, the checklist was adopted by the planning group, and used during planning as the central overview of the individual artists, and used as the central record when information was handed over to the stage groups.

The checklist had three functions: as a tool for the collection of information during planning, secondly as a medium for forwarding information from the group responsible for technical planning to the stage personnel and, finally, as a tool at the stages when receiving the artists and carrying out the performances.

In the design project, the checklist was first transformed into the database sketch by the technical planning group; the checklist became a local design artefact. For the technical planning group this artefact was an incarnation of a technological vision, and it was a reminder of pre-production work, and of how this work supports the production at the stages. For the designers the database sketch was first a too narrow technical vision, but later, during a design workshop, it became an almost complete specification for a prototype.

The researchers had previously made a OMT-like (Rumbaugh et al., 1991) object-oriented analysis of the festival, but this was put aside at the point when the checklist came to act as design specification. The object-oriented descriptions were too unfocused to make sense in the workshop with the users, and they were not needed from a software perspective since we used a DBMS that offered sufficient structuring. The object-oriented analysis' primary function was as a condensed memory for the designers.

The original checklist as it was made by one of the stage groups was a crystallisation of festival work. They were doing the same things every year when they received the files from planning, and when the artists arrived at the festival. They made notes on sheets of paper and these notes gradually became more standardised and, in the end, the pre-printed checklist was made. Thus knowledge about how to receive the artists and what to look out for was embedded in the checklist. The use of the files without checklists was crystallised into the new artefact, the pre-printed checklist.

When the technical planning group took over the checklist, it was transformed from a local artefact supporting work in one specific stage group, into a general artefact used across different groups in the festival organisation. In this "new" form the checklist served a broader range of functions, it became a planning tool and a media for information exchange. The checklist became a boundary object having different meanings for the different groups using and recreating it.

In the design project the researchers understood the developed prototype as the new checklist, with which the technical planning group could do everything they previously did with the paper checklist. An important aspect of the paper-based checklist was that it was handed over to, and used by the stage groups. The way this was done with the prototype was that the information in the database was printed out on paper and attached to the band file. In this way, the reincarnated checklist returned to the stage groups, but for them it was not a checklist anymore, because it had become a mere printout of the pre-production database. The boundary objectness of the checklist was destroyed.

An object-oriented approach to support the transformation process in the festival project, would have focused on modelling (historical stages of) the checklist and possibly other artefacts existing in the festival. Then it would

have analysed how different activities were mediated by these artefacts, how they were similar, how they differed, and which problems existed. Based on this it would have been possible to transform artefacts from the festival through continued crystallisation of acts with the artefacts into the next version.

Physical modelling and Creativity

Understanding design in terms of transformation of artefacts is a vehicle for insuring that the artefacts being designed come to fit the praxis they are intended for. Transformation is in a way an imitation of how the notion of crystallisation tells us that artefacts in a praxis develop. However, understanding design as transformation of artefacts also limits the innovative aspects of design, in some sense prohibiting the greater quantum leaps. The dilemma is that we want to maintain the praxis in question, but we do not want it to stay the same. Thus, in addition to *transformators*, design artefacts mediating design as a transformation process, we need *generators*, design artefacts mediating creation of the radically new. Examples of generators are future workshops (Jungk & Müllert, 1987), and springboards (Engeström, 1987). The concept of tertiary artefacts (Wartofsky, 1973), i.e. artefacts that mediate the creation of autonomous rooms for authentic creation, not related to the productive praxis in an obvious way, is close to the concept of generators.

Object-oriented design artefacts have a potential role as generators. This is one of the main elements in the APLEX (Bødker et al., 1988), a vision of an object-oriented environment for participatory design. Object-oriented design artefacts are likely to enhance the participation of non-computer professionals in the creation of technical visions, because modularity and encapsulation make it possible to build software like kids build LEGO models, and because of a shorter time from idea to running prototype. However, the use of object-oriented design artefacts in the generation of visions may amputate the process by enforcing a too narrow focus on information and computer technology per se.

Conclusions and discussion

Above it is argued that object-oriented methods in the design of computer artefacts can support the involvement of users in the design process, if the

prevailing naturalism in the literature on object-oriented methods is replaced with an activity theory based understanding of the use of representations and models in human praxis.

The key to a new understanding of object-oriented methods is to realise that representations are imbedded in human praxis, and mediating the continuous reproduction of praxis. They are not only second best pictures of the unavailable, or more transparent selections of interesting features of some part of the world, but tools mediating action in the world. Representations are *not* like maps with a suitable detail and selection of features from a chosen terrain, but rather place holders for directions of the continual development of given praxes.

Object-oriented descriptions are intended to be translatable into machine executable code in an unambiguous way, but at the same time they have to serve as ambiguous instruments in the creation of new technological visions. However, this is not different from the boundary objectness of other artefacts mediating heteropraxiality, but the possibility of losing the plasticity is obvious. In this respect, we should try to maintain the so-called CASE-gab.

The use of object-oriented design artefacts is in itself not enough to ensure usability and appropriateness of the artefacts being designed. Thus, I suggest that future object-oriented development methods integrate notions of heterogeneity. To maintain the boundary objectness of representations in design, representations and models should be understood in terms of transformation of artefacts rather than models of domains, thus making possible the continuous confrontation, during design, of the representation with the reality of praxes involved.

Acknowledgement

Thanks to Poul and other people from the music festival who took part in the project; and to Henrik Bærbak Christensen, Preben Mogensen, Kim Halskov Madsen, and Morten Kyng, who were the participating researchers.

References

Bannon, L. & Bødker, S. (1991), *Beyond the Interface: Encountering Artifacts in Use*, in Carroll, John M. (ed.) *Designing Interaction*, Cambridge: Cambridge University Press.

Bannon, L. J. & Kuutti, K. (1993). Searching for Unity among Diversity: Exploring the "Interface" Concept, In Ashlund, Stacey et al. (eds.), *INTERCHI '93: Conference on Human Factors in Computing Systems INTERACT '93 and CHI '93 Bridges Between Worlds*. New York, N.Y.: ACM Press.

Bertelsen, Olav W. (1996a), Contradictions in the Festival Project - Activity systems, obstacles and dynamic forces in design, in Dahlbom, Ljungberg, Nuldén, Simon, Sørensen & Stage (eds.), *Proceedings of the 19th Information systems Research seminar In Scandinavia, (IRIS 19), 10-13 August 1996, at Lökeberg, Sweden*.

Bertelsen, Olav W. (1996b) The Festival Checklist: design as the transformation of artefacts, in *Proceedings of Fourth Biennial Conference of Participatory Design*.

Bjerknes, Gro, Pelle Ehn, & Morten Kyng (eds.) (1987), *Computers and Democracy*, Aldershot UK: Avebury.

Booch, G. (1994) *Object-Oriented Analysis and Design with applications*, 2nd ed. Redwood City, CA: Benjamin/Cummings.

Bærentsen, Klaus (1989), *Menneske og maskine* [Man and Machine], in Hedegaard, Hansen & Thyssen (eds.), *Et virksomt liv* [An active life], pp. 142-187, Aarhus: Aarhus Universitets Forlag.

Bødker, S. & Grønbæk, K. (1996), Users and designers in mutual activity: An analysis of cooperative activities in system design, in Engeström, Y. & Middleton, D. (eds.), *Cognition and Communication at Work*, Cambridge MA.: Cambridge University Press.

Bødker, S., P. Ehn, J. Kammersgaard, M. Kyng, & Y. Sundblad (1987), A UTOPIAN Experience: On Design of Powerful Computer-Based Tools for Skilled Graphic Workers, in Bjerknes, Gro, Pelle Ehn, & Morten Kyng (eds.), *Computers and Democracy*, Aldershot UK: Avebury.

Bødker, S., Knudsen, J.L., Kyng, M., Ehn, P., Madsen, K.H.. (1988) Computer Support For Cooperative Design, in *Proceedings of CSCW '88*.

Coad, Peter & Edvard Yourdon (1990), *Object-Oriented Analysis*, Engelwood Cliffs N.Y.

Ehn, Pelle & Kyng, Morten (1987) The Collective Resource Approach to System Design, in Bjerknes, Ehn, & Kyng, *Computers and Democracy*, Aldershot: Avebury.

Ehn, Pelle & Dan Sjögren (1991), From system description to script for action, in Greenbaum, Joan and Morten Kyng (eds.), *Design at work: cooperative design of computer systems*. Hillsdale: LEA.

Engeström, Yrjö (1987). *Learning by expanding: an activity-theoretical approach to developmental research*. Helsinki: Orienta-Konsultit Oy.

- Goldberg, Adele & Rubin, Kenneth S. (1995), *Succeeding with objects: decision frameworks for project management*, Reading, MA: Addison-Wesley Publishing Company, Inc.
- Greenbaum, J. & Kyng, M. (eds.) (1991), *Design at Work*, Hillsdale, N.J.: LEA.
- Holbæk-Hansen, E., Håndlykken, P., & Nygaard, K. (1975) *System Description and the Delta Language*, Oslo: Norwegian Computing Center.
- Jackson, Michael (1983), *System Development*, Engelwood Cliffs, N.J.: Prentice-Hall.
- Jacobson, I., Cristerson, M., Jonsson, P. & Övergaard, G. (1992), *Object-Oriented Software Engineering—A Use-Case Driven Approach*, Reading MA: Addison-Wesley.
- Jungk, R. & Müllert, N. (1987), *Future Workshops: How to create desirable futures*. London: Institute for Social Inventions.
- Leontjev, A. N. (1978). *Activity, consciousness, and personality*, Engelwood Cliffs NJ: Prentice Hall.
- Madsen, Ole Lehrmann, Møller-Pedersen, Birger & Nygaard, Kristen (1993) *Object-Oriented Programming in the BETA Programming Language*, Wokingham, England: ACM Press/Addison-Wesley Publishing Company.
- Madsen, K.H. (1996) Initiative in participatory design in *Proceedings of Fourth Biennial Conference of Participatory Design*.
- Mathiassen, Lars (1981), Systemudvikling og Systemudviklingsmetode [in Danish: System development and system development method], Aarhus: Computer Science Department DAIMI PB-136.
- Polanyi, Michael (1966), *The Tacit Dimension*, Garden City, NY: Doubleday.
- Rumbaugh, James, M. Blaha, W. Premerlani, F. Eddy, & W. Lorensen (1991), *Object-Oriented Modelling and Design*, Prentice-Hall International Editions.
- Star, Susan Leigh (1989), The Structure of Ill-Structured Solutions: Boundary Objects and Heterogeneous Distributed Problem Solving, in Gasser, Les and Michael N. Huhns *Distributed Artificial Intelligence, volume II*, London: Pitman Publishers, pp. 37-54.
- Vygotsky, L.(1978). *Mind in society: The development of higher mental processes*, Cambridge, MA: Harvard University Press.
- Wartofsky, Marx W (1973), Perception, representation, and the forms of action: toward an historical epistemology, in Wartofsky, M. W., *Models*, Dordrecht: D. Reidel Publishing Company, 1979.
- Yourdon, Edvard (1982), *Managing the system life cycle*, N.Y.: Yourdon Press.

References to included papers

- 1: Bertelsen, O. W. (1994a). An investigation into the design artefact concept as a vehicle for systems development research. In *Proceedings of the 17th Information systems Research seminar In Scandinavia IRIS 17, Syöte, Finland, August 1994*. pp. 715-125.
- 2: Bertelsen, O. W. (1994b). Fitts' Law as a Design Artefact: A Paradigm Case of Theory in Software Design. In Blumenthal, Gornostaev, & Unger (eds.). *Human-Computer Interaction. 4th International Conference, EWHCI '94 St. Petersburg, Russia, August 1994. Selected Papers*, Berlin: Springer Verlag. pp. 11-18.
- 3: Bardram, J. E. & O. W. Bertelsen (1995). Supporting the Development of Transparent Interaction. In Blumenthal, Gornostaev, & Unger (eds.): *Human-Computer Interaction. 5th International Conference, EWHCI '95 Moscow, Russia, July 1995. Selected Papers*. Berlin: Springer Verlag (LNCS 1015). pp. 79-90
- 4: Bertelsen, O. W. (1996a). Contradictions in the Festival Project - Activity systems, obstacles and dynamic forces in design. In Dahlbom, Ljungberg, Nuldén, Simon, Sørensen & Stage (eds.), *Proceedings of the 19th Information systems Research seminar In Scandinavia, (IRIS 19), 10-13 August 1996, at Lökeberg, Sweden*. pp. 597-612.
- 5: Bertelsen, O. W. (1996b). The Festival Checklist: design as the transformation of artefacts. In Blomberg, J., Kensing, F. & Dykstra-Erickson (eds.). *PDC '96, Proceedings of the Participatory Design Conference*, Palo Alto: Computer Professionals for Social Responsibility. pp. 93-101.
- 6: Bertelsen, O. W. (1996c). Organisational learning is crystallised into artefacts [Position paper for workshop on organisational learning at CSCW'96]. In *SIGOIS Bulletin* vol. 17, no. 3, December 1996. pp. 37-39.
- 7: Bertelsen, O. W. (1997a). Understanding objects in use-oriented design, in Braa, Kristin & Monteiro, Eric (eds.). *Proceedings of the 20th Information systems Research seminar In Scandinavia, Oslo, 1997*. pp. 311-324.