Scandinavian Journal of Information Systems

Volume 1 | Issue 1 Article 3

1989

THE ART AND SCIENCE OF DESIGNING COMPUTER ARTIFACTS

Pelle Ehn

University of Aarhus, Denmark, PelleEhn@emailaddressnotknown

Scandinavian Journal of Information Systems, Vol. 1, pp. 21–42, 1989

THE ART AND SCIENCE OF DESIGNING COMPUTER ARTIFACTS

PELLE EHN

Department of Information and Media Science at University of Aarhus
Niels Juelsgade 84, DK-8200 Århus N, Denmark

Abstract

This is a paper about activities such as information system development and software engineering. The specific aspects of these activities that are being investigated are the academic organization for these studies, and the doctrines that are being taught at departments of computer science, information science, informatics, etc.

The paper begins with a critique of the existing division of labour in the academic field. The need to transcend the dominating natural science oriented tradition in design of computer artifacts is argued. To replace this tradition a new foundation guided both by a technical knowledge interest in instrumental control as in the natural sciences *and* a practical knowledge interest in inter-subjective communication as in the social sciences and the humanities is suggested.

It is argued that the dominating doctrine of a *rationalistic science of the artificial* is too limited, and that it ought to be replaced, or complemented, by a doctrine for a *practical art and science of designing computer artifacts.*

Finally, a teachable programme for a disciplinary base of and art and science of designing computer artifacts is outlined.

Keywords: design, computers, artifacts, systems, software, organization, education, systems development, systems design, computer science, information science, informatics, art, science.

Both art and design at last seem like meeting,
across the Cartesian split of mind from body,
to enable us to find a new genius collaboration
not in the making of
products and systems and bureaucracies
but in composing of contexts that include everyone,
designers too.
To be a part.
To find how to make all we do and think
relate to all we sense and know,
(not merely to attend to fragments
of ourselves and our situations).
It was a question of where to put your feet.
It became a matter of choosing the dance
Now its becoming
No full stop

J. Christopher Jones in
How My Thoughts about Design Methods have Changed During the Years

1 Introduction

1.1 Design of Computer Artifacts

This paper concerns activities that go under such names as system design, system development, systemeering, software engineering, etc. These activities are here referred to as *design of computer artifacts* in an attempt to avoid taking for granted some strongly embedded rationalistic presumptions of this activity.

I understand design of computer artifacts as a concerned social and historical activity in which artifacts and their use are envisioned, an activity and form of knowledge that is both planned and creative, and that deals with the contradiction between tradition and transcendence. The use of artifacts and the users themselves, not only the designer and the artifact in insolation, become constituent aspects of the design process (Ehn 1988).

From this point of view I will investigate a very specific aspect of design of computer artifacts—*the academic organization for studies of design and the doctrines that are being taught.* This aspect of design and computers is here being referred to as *the art and science of designing computer artifacts*, in an attempt to avoid taking for granted the boundaries between art and science, and between natural, social and human sciences, manifested in the ruling paradigm of rationalistic natural science in computer studies.

1.2 1968—The Spell of Tradition

To me, 1968 stands out as a remarkable year in the history of the art and science of designing computer artifacts. Not because that was the year when I started my studies in Information and Computer Science at the University of Gothenborg. Three other events are far more interesting.

First of all, the years around 1968 was the time when *the first academic departments* for our discipline were established in Scandinavia. For an historic overview see (Bansler 1987).

Secondly, 1968 was also the year when Noble Prize winner Herbert A. Simon in a famous lecture at M.I.T. outlined *a programme for a rationalistic science of design*, and a curriculum for training of professional designers. He did this with engineering as an example, but he also argued that other fields were just as relevant, for instance, management science and computer science, fields from which he himself had experience (Simon 1969). This was a programme with far reaching consequences for the development of our art and science of designing computer artifacts.

Thirdly, 1968 was the year of the *student revolt* and the beginning of a process that aimed to democratize our academic institutions and to make scientific knowledge useful to ordinary people. A process that had strong influence on my life and more significantly the formation of the critical Scandinavian approach to systems development, an attempt to democratize design and use by the influence from trade unions and their members (Ehn & Kyng 1987).

These events of 1968 were responses to different crises. Departments of mathematics, engineering or business administration were found to be insufficient platforms for the study of information and computer science. Hence, new departments. The engineering approach to design of the artificial was declared to be a tradition bound craft not a real science. Hence, the rationalistic programme for a real science of the artificial. The political rationality of science was considered counter-productive to the interests of democracy and emancipation. Hence, the student revolt and the 'collective resource' approach.

In different ways these three events are still alive, influencing our academic life in Scandinavia. The early organization, with departments built around people with a natural science or mathematical background, focusing on the computer as such or on abstract information processing, still literally "inform" our activities. The same is the case with the rationalistic programme for a science of design of the artificial that Simon informed us with. However, the 'collective resource' approach to democratization has also survived.

A few years ago I started to write a doctoral dissertation about my 20 years of travels in the research landscape of information and computer studies in Scandinavia. My interest during these years are reflected in the title of the thesis: *Work-Oriented Design of Computer Artifacts* (Ehn 1988). The book is an investigation into the practical and theoretical possibilities of designing for democracy and skill. However, in the thesis work I also had to search for an academic 'home'

for design of computer artifacts. So when the editors of the *Scandinavian Journal of Information Systems* generously offered me to contribute with a paper based on my thesis for the first issue of the journal I decided to focus on the more general "home" topic and the foundations that we have inherited from the late sixties, rather than on my own 1968-tradition and the 'collective resource approach' to work-oriented design. Hence, the focus of the paper will be on *the rationality of the academic organization of information and computer studies, and the doctrines that we teach.*

1.3 Structure of the Paper

I start out with the critique of our *academic organization* of computer and information studies. I am not concerned with the names of academic departments conducting such studies, whether they should be called computer science, information science, information processing, informatics, datalogy or whatever, but on what we should be doing in research and in teaching in the art and science of designing computer artifacts. I argue that we must transcend the prevailing division of labour between our academic disciplines, focusing instead on the subject matter itself, reconstructing it so that we can grasp *use* as a fundamental aspect of design. In search for directions for reorganization I then turn to a discussion of the *a priori* anthropological everyday *knowledge interests* that guide programmes and organization of our academic activities.

In the second part of the paper I draw the readers attention to our taken for granted background—the rationalistic tradition. Herbert Simon's programme for the *rationalistic science for design of the artificial* will be used as example, but it is suggested that the early programme for software engineering, and the infological approach in Scandinavia are other examples of the same tradition. Against this background a reformulation of the subject matter is outlined, and I sketch a complementary programme for an *art and science of designing computer artifacts*, and a curriculum for its study.

2 Academic Organization

2.1 Institutional Boundaries

Design of computer artifacts is not only studied at natural science departments, but also in social sciences and in the humanities. Nevertheless natural science based computer science departments are still often considered as t*he real place for information and computer studies.* So, I will make computer science my point of reference. What is computer science? Not long ago this question was addressed by Paul Abrahams, president of the Association for Computing Machinery, in *Communications of the ACM* (Abrahams 1987).

As a first conclusion he suggests that any definition whose scope strays too far from the pragmatic answer that computer science is what is taught by computer

science departments is unlikely to meet much acceptance. I guess he is right. Nevertheless I agree with him that the question is worth discussing.

I also agree with Abrahams that computer science is not the study of Vaxes or Macintoshes, not even of Connection Machines or Turing Machines, though such studies may be part of it. However, I am afraid that this is where the agreement stops. At least when he suggests that most of the interesting questions in theory are special cases of two general types: what can we compute, and what resources do we need in order to compute it? I can imagine many other theoretically interesting questions for an art and science of designing computer artifacts. These questions concern designing computer artifacts for concerned human use. The disagreement may be due to a different emphasis, but I think it goes deeper.

It is not a disagreement on the belief that theory of computer science includes such specialities as algorithmic analysis, computational complexity, and formal language theory, but perhaps on the scope of what we are studying, and what other theories we need. It is not a disagreement on the position that pragmatic computer science has a flavour of engineering (or is concerned with design of computer artifacts, as I would put it) but perhaps on the theoretical consequences of this position. Finally, it is not a disagreement on the position that the micro-structure of computing is inherently mathematical, and that the macro-structure may not be, but maybe on what it may be.

I have neither competence nor reason to challenge the mathematical and natural science base of computer science when the subject matter is efficiency of algorithms, semantics of programming languages, computability, etc. However, I will have to consider it problematic as soon as any kind of human use of computers is involved, or when any social or organizational setting for its design ought to be taken into account. Here I am primarily thinking of systems development, but I also think matters become problematic in subject areas such as knowledge-based systems, human-computer interaction or design of programming environments.

In fact, the idea of mathematics and natural science as normal science (Kuhn 1962) for a science of designing computer artifacts is due to history, tradition and coincidence, rather than fundamental reflections of the subject matter. Here, I have the history of our academic institutions in Scandinavia in mind (Bansler 1987). When the first departments were established in the late sixties it was typically around people with a mathematics or natural science background. They had either participated in constructing computers or used computational power in their academic work. The focus on natural science and the neglect of other scientific perspectives may have been reasonable then. Today such a focus is too narrow, especially if systems development is to be part of what is studied at such departments. One obvious reason is the tremendous expansion in use of computers during the two decades when departments for computer or information science have existed in Scandinavia. Still, the natural science based tradition from 1968 seems hard to transcend. Why? What arguments towards change could be made? Personally, I have met two kind of arguments, and in my opinion none of them holds.

The first arguments is based on *the need for division of labour.* The argument is that computer science only deals with the natural science and mathematical aspects of computers, the rest is being left to other disciplines in the human and social sciences.

I think that this argument holds in so far as the effects of using computers are studied by many other disciplines. However, there is, to my knowledge, no computer science department in Scandinavia that does not at least have something like knowledge-based systems, human-computer interaction, design of programming environments, or systems development in their curriculum for the students. Is the study of these subjects true natural science? I question the fruitfulness of such an assumption. I will also, at least in the case of systems development, argue against the fruitfulness of the prevailing division of labour between our academic disciplines. One of my arguments against this academic division of labour, an argument that can be raised as a critique from inside computer science, is that many researchers and teachers in the human and social sciences base their statements on too limited an understanding of computers. My other argument against the prevailing division of labour is more fundamental. Social and human sciences play an important role in the study of effects, or long range social consequences of adaptation of computers in society. However, their role in *design* of computer artifacts is so far very limited and certainly constrained by the existing division of labour between the disciplines. When it comes to systems development I see this as a major obstacle.

Yes, we need division of labour in our academic field, but the existing division of labour is a dysfunctional, historic reminiscence. *An organizational change is needed.*

The other argument that I have met against change, explicitly include systems development in computer science at the same time as human and social science approaches are excluded. It is the idea of *a science of design of computer artifacts as engineering based on natural science theories and methods.* Historically I see this as the main approach to systems development. This approach seems stronger than ever today. A good example is the recent plea for real systems engineering by Janis Bubenko, a leading Scandinavian professor in our field. In a conference invitation recently he formulated the theme like this: "Information Systems Engineering represents an approach to information systems development that is based on an 'engineering' way of coming to grips with the different tasks to be solved in large systems development projects. (...) Information Systems Engineering represents work carried out by 'engineering' analysts and constructors in a rigorous, methodological way of coming to grips with every sub-problem, no task in the systems development being solved by capricious approaches" (Bubenko 1987, my translation).

Yes, we need a theoretically sound foundation for design of computer artifacts, but to solely base this on an engineering approach from the natural sciences is dysfunctional. *A change of basic doctrine is needed* To explain why I take this position the notion of *knowledge interests* will be introduced.

2.2 Knowledge Interests

The notion of *knowledge interests* has been developed by the social philosopher Jürgen Habermas (Habermas 1968a and 1968b). According to Habermas these *a priori* anthropological everyday procedures and interests of knowledge determine the conditions under which every science objectifies reality. He distinguishes between a *technical, a practical*, and an *emancipatory* knowledge interest.

The *technical control interest* leads to focus on observation, empirical analyses, and instrumental control, as in the technical and natural sciences. This is at the core of a science of design of computer artifacts concerned with the purposive rational design of systems, and the technical functionality of these systems. This is the aspect of design where we as human beings, encounter objects as things, events, and conditions which, in principle can be manipulated.

The *practical interest in inter-subjective communication* leads to focus on dialogues, participatory relations, and understanding, as in the social sciences and the humanities. If we accept that design and use of computer artifacts also are historical and social processes then we have to be concerned with this practical knowledge interest in inter-subjective communication. For an art and science of designing computer artifacts this means a focus on interpretation, and human communication, and the establishment and expansion of action oriented understanding.

The dilemma in determining the subject matter for an art and science of designing computer artifacts is that both the technical knowledge interest in instrumental control, and the practical knowledge interest in inter-subjective communication, seem equally fundamental. Furthermore, these double knowledge interests are fundamental both to *design* and to *use*. Design of computer artifacts is an activity of determining these artifacts so that they can be constructed and implemented. Hence, the technical interest in instrumental control. But it is also a dialogue and a participatory relation between those concerned about the computer artifact being designed. Hence the practical interest in inter-subjective communication. Considering the use situation designed for, there is the same doubleness. Computer artifacts may be designed to support control of objects as well as to facilitate dialogues and inter-subjective communication.

This doubleness I see as the fundamental condition under which design of computer artifacts must be objectified as a scientific subject matter. Hence, in studying this subject matter we must transcend the disciplinary boundaries between the natural sciences, the social sciences and the humanities, to be able to deal with the different interests of knowledge that constitute the subject matter. The practical knowledge interest cannot be abandoned to *a posteriori* studies by the human sciences and the social sciences, because *the design process is where the action is.* Critique may help change the conditions for design and use, but not until integrated into theory and methods of design can these interests have any real impact on how people design and use computer artifacts.

Finally, the *emancipatory interest* is a consciously incorporated interest in sci-

ence that directs knowledge towards emancipation going beyond the other interests of knowledge. As paradigm examples Habermas mentions the critique Marx developed as a theory of capitalism, and Freud as a meta-psychology. This is the interest in the process of critique as a means to reveal power relations embodied in our socio-cultural form of life as systematically distorted communication—the interest of liberation and a dialogue free from coercion through knowledge. Hence the fundamental relations to political practice in Marxist theory and to therapy in Freudian theory. I am not proposing that the emancipatory interest of knowledge must be constituent to all aspects of the subject matter of designing computer artifacts, though the fundamental relation between such a science and changes in work and language is an obvious argument. However, for the 'collective resource' approach I come from, this interest is cardinal.

In summary: The academic division of labour between natural science, social science and the humanities that we have inherited from the late sixties is a tradition that has not been able to adapt organization and theory to the drastically changed environment. Especially the dominating role of mathematically oriented natural science based computer science departments stands out as problematic. In rethinking the division of labour and the theoretical orientation Habermas notion of knowledge interests can be most useful. In an art and science of designing computer artifacts we need organization and theory that helps us deal *both* with the technical interest of instrumental control and the practical interest of inter-subjective communication. Maybe even the interest of emancipation can be regarded as a legitimate research guiding interest once we transcend our rationalistic tradition in design of computer artifacts.

3 The Rationalistic Tradition

In the introduction I suggested that Herbert Simon's programme for a rationalistic science of design of the artificial (Simon 1969) maybe the most important influence from 1968 on our tradition in the art and science of designing computer artifacts.

As students, we have since the late sixties, met Simon's programme or the rationalistic tradition in many guises.

Some of us have been confronted with the branch of rationalistic design formulated as the first programme for software engineering (Naur & Randell 1969). This was the idea of understanding computer programs as formal mathematical objects derived by formalized procedures from an abstract specification. The correctness of programs was guarantied by methods for successive transitions from specification texts to computer-executable code. This development was very much a reaction to the software crisis of the late 1960's caused by the advent of powerful third generation computers with complex operating systems, and numerous new sophisticated applications in many fields. The old methods and approaches were simply insufficient. However, this "new" product-oriented view leaves the relationship between programs and the living human world entirely open.

Other of us have met the rationalistic tradition in the form of "theoretical analysis of information systems" (Langefors 1966). This was the idea of defining the elementary abstract information needs in an organization, and then redesign an optimal or at least satisfactory ('satisficing behaviour' as Simon puts it) information system. Based on this information systems analysis, specifications for implementation of computer-based information systems could be derived. This development was a reaction to the confusion of programming and data-structures with information needs in organizations. However, this infological approach remained entirely within the rationalistic realm. The relation between people in an organization and their information needs was a question of objective facts to be discovered by the analyst.

In one or another of its different guises or disguises, the rationalistic tradition of 1968 really is *the* background for most of us educated at some department in Scandinavia, even if oppositional perspectives also have been suggested. To mention a few but significant examples see e.g. the doctoral dissertations by (Ivanov 1972), (Mathiassen 1981), and (Lyytinen 1986).

3.1 The Science of Design of the Artificial

In this paper I have chosen Simon's programme for a science of design of the artificial as point of reference since it is such an explicit and elegant formulation of the rationalistic doctrine of design. Simon suggest that "the proper study of mankind is the science of design, not only as the professional component of a technical education but as a core discipline for every liberally educated man" (Simon 1969, p. 83). He based this statement on his experiences from having lived close to the development of the modern computer and the growing communication among intellectual disciplines taking place around the computer. The ability to communicate across fields does not come from the computer as such, he argued, but from the need to "be explicit, as never before, about what is involved in creating a design and what takes place while the creation is going on." (Simon 1969, p. 83).

He also observed that in order to gain academic respectability, e.g. "engineering schools have become schools of physics and mathematics; medical schools have become schools of biological science; business schools have become schools of finite mathematics. The use of adjectives like 'applied' conceals, but does not change, the fact" (Simon 1969, p. 56). He did argue that this way of acquiring a scientific subject matter had moved the design disciplines away from their real subject—*the design of the artificial.*

On the one hand, Simon took the position that what was traditionally known about design was "intellectually soft, intuitive, informal, and cookbooky" (Simon 1969, p. 57), and that this was scientifically unsatisfactory. At the same time he claimed that design of the artificial is really what most professions are about. Thus it cannot be replaced by mathematics, physics etc. This understanding of a fundamental dilemma for design sciences is an important observation by Simon.

The alternative that Simon suggested is a science of design of the artificial, a genuine science of its own. His elegant solution is to pose the problem of design of the artificial in such a way that we can apply methods of logic, mathematics, statistics etc., just as we do in the natural sciences.

According to Simon's science of design of the artificial, computers are complex hierarchical systems, as are the users and the use organizations, and as is the design process—together they are sub-systems of a bigger system, and we can define their various functionalities separately. In designing these systems we can use many scientific methods (based on theory in formal logic, mathematics and statistics) for evaluation of designs and in the search for alternatives.

About computer artifacts the science of design of the artificial informs us that computers are systems "of elementary functional components in which, to a high approximation, only the function performed by those components is relevant to the behaviour of the whole system" (Simon 1969, p. 18).

Scientifically they can be studied both as abstract objects by mathematical theory and as objects in an empirical science of the behaviour of computers as complex systems. At the same time the computer artifact is also seen "as a tool for achieving a deeper understanding of human behaviour" (Simon 1969, p. 22). This is, he argued, because of its similarity in organization of components with the "most interesting of all artificial systems, the human mind" (Simon 1969, p. 22).

A science of design of the artificial did, according to Simon, already exist in the late sixties, "particularly through programs in computer science and 'systems engineering" (Simon 1969, p. 58). Management science was also included among the systems or design sciences that had started to develop "a body of intellectually tough, analytic, partly formalizable, partly empirical, teachable doctrine about the design process" (Simon 1969, p. 58).

To Simon the natural sciences are concerned with how things are, whereas design is concerned with how things *ought to be*—we devise artifacts to attain goals. However, Simon saw no need for a new logic to deal with the normative character of design. The problem of design can be reduced to declarative logic. Hence, the teachable doctrine Simon suggested had the following seven topics (Simon 1969, p. 62–79):

1. Utility theory and statistical decision theory—a logical framework for rational choice among given alternatives (...)

2. The body of techniques for actually deducing which of the available alternatives is the optimum (...)

3. Adaption of standard logic to the search for alternatives (...)

4. The exploitation of parallel, or near-parallel, factorizations of differences (...)

5. The allocation of search resources to alternative partly explored action sequences (...)

6. The organization of complex structures and its implication for the organization of the design process (...)

7. Alternative representations for design problems.

It is beyond the scope of this paper to give an elaborate account and critique of the seven topics. Instead I will make a few summarizing comments.

To accept Simon's rationalistic programme we have to assume that design is a process of *problem-solving by individual decision-makers among sets of possible worlds.* It may be that this transforms the question of design into the rationalistic scientific vein, but at the same time most essential aspects of design are lost. I am thinking of the creativity of professional designers and users that by its very nature defies to be reduced to formalized decision-making. I am also thinking of the social and historical character of the design process—the conflicting interests, the differences in skill, experiences and professional languages. Given such aspects, not much in the science of design of the artificial seems useful in organizing the design process, and in designing in a social and historical setting.

3.2 From the Artificial to the Practical

The science of design of the artificial leads us in the direction of what we today know as the discipline of artificial intelligence. And even though computer artifacts are in the centre of most of Simon's examples of a science of design of the artificial, we are not told how to design computer artifacts that can be skillfully used by concerned users.

Simon's rationalistic science of design of the artificial may be an ever so "intellectually tough, analytic, partly formalizable, partly empirical, teachable [a] doctrine about the design process" (Simon 1969, p. 58). Still we are forced to question his argument that since we have explicit knowledge of the design process in computer programs running optimizing algorithms, search procedures, etc., there is no need to retreat to "the cloak of 'judgment' or 'experience"'. (Simon 1969, p. 80). Practical understanding that shrinks when transformed by formalisms and put into the computer may well be the most essential experience and judgement in professional design.

In short: Why should we accept that problem decomposition is more fundamental than problem identification, that descriptions should focus on redundancy rather than on uniqueness, that reduction of complexity by decomposition into simple sub-systems is more relevant than to critically consider wholes, that subjectivity should be avoided rather than included in our understanding of objectivity? Why are, as Werner Ulrich has put it in a fictitious Simon-Churchman debate, "semantic precision of concepts, model building, explanation, mathematical analysis, empirical research, computer simulation, heuristic programming,

scientific rigor and programmed decision making" more important to the designer than, "reflection on the sources of knowledge and deception, ideas, experience, imagery, affectivity, faith, morality, interest, on-going debate and self-reflection in order to unfold problems and conflicts" (Ulrich 1980, p. 38).

In my thesis (Ehn 1988) I try to demonstrate the fruitfulness of an understanding of design of computer artifacts in a language that transcends the rationalistic natural science based language of systems, objects, information, and data. I also argue that with such an understanding of the subject matter, there already exist well elaborated and teachable theories and doctrines about what professional designers should do and know.

However, I do not argue for a re-invention of the wheel: the instrumental power of systems thinking for purposive rational action is beyond doubt, and many of the computer applications that function well today could not have been designed without rationalistic design methods. Instead I suggest a re-interpretation of design methods to take us beyond the so strongly embedded Cartesian mind-body dualism and the limits of formalization, towards an understanding that hopefully can support more creative designer ways of thinking and doing design as cooperative work, involving the skills of both users and designers.

The design approach that I outline is an attempt to include subjectivity in a double sense. I claim the importance of rethinking the design process to include structures through which ordinary people at their workplace more democratically can promote their own interests. I also claim the importance of rethinking the use of descriptions in design, and of developing new design methods that enable users of new or changed computer artifacts to envision their future use situation, and to express all their practical competence and creativity in designing their future.

Based on this understanding I will below outline a curriculum for a disciplinary base in the studies of the art and science of designing computer artifacts.

4 The Art and Science of Designing Computer Artifacts

My arguments so far should hopefully have convinced the reader that we really need a complementary programme in the art and science of designing computer artifacts. The good news is that there already exists well elaborated and teachable theories and doctrines to include in such a programme. We do not have to start from scratch.

To paraphrase Simon, my thesis like his programme suggests seven topics. They are:

1. Social system design methods.
2. Theory of designing computer artifacts.
3. Process-oriented software engineering and prototyping.
4. History of design methodology.

5. Architectural and industrial design as paradigm examples.

6. Philosophy of design.

7. Practical design.

Below follows in summary my seven candidates for inclusion in a curriculum to be taught in disciplines dealing with the art and science of designing computer artifacts. For each of the topics I have included a list of suggested readings. Neither the choice of topics, nor the lists of suggested readings, are intended to be exclusive, but should rather be understood as a possible start. Suggestions for improvements are more than welcome.

4.1 Social System Design Methods

This topic covers methods which include *the role of subjectivity* in design of systems. Theoretically the different approaches to social systems design have their origin in rationalistic systems thinking, but transcend this framework philosophically by including the subjectivity of the users. The different approaches are well developed and they are based on extensive practical experience. However, these approaches are fundamentally 'pure' methodology. Hence a challenge will be to investigate how they can be integrated with, or supplemented by, substantial theory of the social situations in which they are to be used. Another challenge is how they can be refined to more specifically deal with design of computer artifacts and their use.

Suggested readings: Here I am especially thinking of three design researchers that have developed challenging alternatives to Simon's systems engineering. They are C. West Churchman, Russell L. Ackoff, and Peter Checkland. To Churchman a designer is engaged in detached reflection over the possibilities for developing and implementing good systems. In this he is guided by the *systems approach.* This is a special development of ideas from American pragmatism, especially Churchman's philosophical mentor E. A. Singer. The systems approach is developed in (Churchman 1968). The philosophical background is dealt with in (Churchman 1971). Ackoff is another of Singer's disciples who has had a major influence on social systems thinking in operations research. See e.g. (Ackoff 1974). As with Churchman, he includes subjectivity in objectivity. But whereas Churchman is basically interested in ideas, Ackoff argues the crucial role of *participation.* To him objectivity in design is the social product of the open interaction of a wide variety of individual subjectives. Ideally the design process involves as participants all those who can be directly affected by the system, its stakeholders. Ackoff's designer does not like a doctor identify or solve organizational messes by diagnoses, or prescriptions. He is more like a teacher than a doctor. The designer is someone that through encouragement and facilitation enables the participants and stakeholders to deal more effectively with their organizational messes—and having fun doing this. Checkland finally, like Ackoff and Churchman, started out

in the tradition of rationalistic systems engineering. Like them he found that systems engineering simply was not appropriate for practical intervention in the complex and ambiguous *'soft' problem situations* in social practice. His systems methodology is focusing on the importance of the dialectics between the many and different world views involved. With Checkland a step is also taken away from rationalistic systems thinking towards interpretation and phenomenology. See (Checkland 1981).

4.2 Theory of Designing Computer Artifacts

This topic covers fundamental theory of what kind of phenomenon the design of computer artifacts actually is. It reframes the rationalistic understanding of computer artifacts. The point of departure is what people do with computers in concerned human activity within a tradition. This also means an emphasis on differences in kinds of knowledge between on the one hand human practical understanding and experience, and on the other, knowledge representation in computers understood as 'logic machines' or 'inference engines'.

In design, focus is on concerned involvement rather than on correct descriptions. Design becomes a process for anticipation of possible breakdowns for the users in future use situations with the computer artifacts being designed. This is anticipation both to avoid breakdowns, and to recover from them.

As 'founders' of this tradition two persons have been especially important, philosopher Hubert L. Dreyfus and computer scientist Terry Winograd. Dreyfus argues for the relevance and importance of skill and everyday practice, in understanding the use of computers. His investigations are based on the philosophical positions of existential phenomenology in the tradition of philosophers like Heidegger and Merleau-Ponty, and the positions on ordinary language and language-games taken by Wittgenstein. Winograd has brought this view into computer science and the design of computer artifacts for human use. To Winograd it is in involved practical use and understanding, not in detached reflection, that we find the origin of design. Design is the interaction between understanding and creation.

Being a new and fundamentally ontological approach to design of computer artifacts, not much instrumental design methodology has as yet been developed. Neither has its relation to substantial social theory been extensively investigated. In both respects there are challenging possibilities.

Suggested readings: The two most important books by Dreyfus are (Dreyfus 1972) and (Dreyfus & Dreyfus 1986). Winograd's arguments have been put forward in the by now classical (Winograd and Flores 1986). A good complement to those books is the ethnomethodological approach to plans and situated actions by anthropologist Lucy Suchman (Suchman 1987).

4.3 Process-Oriented Software Engineering and Prototyping

This topic is a paradigmatic rethinking of the process of designing software much along the lines of the first two topics. The paradigmatic shift of primary point of view is from design of software as formal mathematical objects derived by formalized procedures from abstract specifications, towards a process-oriented view. In this new paradigm, the software engineering focus is on human learning and communication in the design process, and the *relevance, suitability and adequacy in practical use situations of the software being designed,* not just on the correctness and efficiency of the piece of code being produced.

The relevance of this view is also accentuated by the development in software engineering of powerful computer-based design artifacts for prototyping. So far this is a very open field including fourth generation tools (typically a database, a query language, a screen editor and a report generator) as well as more advanced exploratory programming environments (like Smalltalk or Lisp/Loops). Other aspects have to do with the kind of prototyping that is supported, e.g. horizontal prototyping (all functions, but with limited functionality) or vertical prototyping (few functions, but with full functionality). There is also the use of early non-computer-based design artifacts like mock-ups to consider.

However, the important point with all these design artifacts is the support for *involved envisionment.* This means a possibility for prospective users in cooperation with professional designers to *gain experience of future computer artifacts* by using these design artifacts as a basis for design requirements. A challenge is to develop programming environments for prototyping that can be integrated with full scenarios, role plays etc., of future use situations, and to use these prototypes as design artifacts in playing language-games of design as games of involvement and doing that defeat some of the limits of formal descriptions. Another challenge is to support the emergence of democratic environments for the utilization of this approach.

Suggested readings: This new orientation in software engineering, which so well match the theoretical perspective in the here outlined programme, is suggested by people that were closely associated with the early software engineering programme. See especially the paradigm shift paper by Chritiane Floyd (Floyd 1987). In fact, Peter Naur, one of the founders of the software engineering tradition have made comments in a similar direction (e.g. Naur 1985). A good introduction to the prototyping field is found in (Floyd 1984). For further references playing the language-games of design see (Ehn 1988). For an introduction to the ideas of involved envisionment and design as co-operative work see (Bødker *et al.* 1988, Ehn 1988, Kyng 1988 and Ehn 1989).

4.4 History of Design Methodology

Another topic for the curriculum in design of computer artifacts is general reflections on design methodology. Computers are not the only artifacts that are designed. *How has design methodology developed in more mature design fields*

such as architectural and industrial design? Why has there e.g. been a shift from rationalistic, formal and mathematically oriented approaches towards both more participatory approaches and more design-like ways of thinking? Why have theoretically influential designers reacted so strongly against their own rationalistic approach "to fix the whole of life into a logical framework" (industrial designer Christopher Jones) that they now even advise us to "forget the whole thing" (architect Christopher Alexander) and start to experiment with art in the design process? It should be important to every well educated designer in our own field to reflect upon the relation between design methodology for computer artifacts and the experiences with different generations of design methodologies in other fields.

Suggested readings: A fine collection of major papers in the 'design methodology movement' have been edited by architect and design researcher Nigel Cross (Cross 1984). Since the early 1960's the movement has developed through three generations. The first rationalistic generation, up to the early 1970's, included important contributions like (Alexander 1964), (Simon 1969) and (Jones 1970). As a reaction to this expert role of the designer a participative second generation design methods developed in the 1970's (Rittel 1984). In the 1980's focus has again shifted, and a third generation trying to understand what designers really do has emerged (Broadbent 1984). The new transcending positions taken by early important members of the design methodology movement can be found in (Alexander 1984) and (Jones 1984).

4.5 Architectural and Industrial Design as Paradigm Examples

Still another relevant topic for our curriculum is *how design is carried out in other design disciplines* like architectural and industrial design. We can use design experiences from these disciplines as paradigm examples to reflect over theory, methods and practice in our own field. This includes reflections over the relations between science and art in design, on styles or 'schools' in design, on the relation between science and styles in design, and on the social relations of designing in complex conflicting and pluralistic social settings.

Maybe these kinds of reflections are not relevant for a *science* of designing computer artifacts in a narrow sense, but certainly for extending it to, and incorporating in it, an *art* of designing computer artifacts, and for our understanding of what kind of enterprise design of computer artifacts is socially. Style might not be scientific, but it certainly plays an important role in design of computer artifacts too. For example, today the 'desktop metaphor' is á la mode in interface design, and 'object orientation' is very popular in programming. What is style or art in this, and what is purely scientific? Ten years from now other styles may be in vogue. It should be worth reflecting about what was science and what was art in the ideas that Alan Kay had, and before him Douglas Engelbart, that today are manifested as Macintosh style workstations? Which ideas led Kristen Nygaard and Ole Johan Dahl to design SIMULA and basic concepts of what

today is known as a school of object oriented programming? As I see it, these innovations are just as much artistic creations of new design styles as new scientific approaches, but that does not make them less important. On the contrary they show the importance of artistic competence in the field of designing computer artifacts.

Certainly an awareness of style, the history of schools and their programs, and experiences with different styles is important in a curriculum in design of computer artifacts. Style may not be scientific in a rigorous sense, but it is professionally important to be able to master different styles. Furthermore, to reflect about styles and schools in design of computer artifacts, as well as to investigate analogies to architectural and industrial design, is a most rational endeavour. After all, design styles and artistic ones live and die as scientific paradigms do. We had better be aware of both, both as tradition and for creative transcendence that can lead to better designs.

Design styles or schools like Bauhaus (as a proactive approach based on the insight that design of artifacts is design of future conditions of living, a vehicle for change) and Postmodernism (as the use of signs and metaphors that are joyful to play with) are good examples of such paradigm cases for reflections over the relations between science, art and society in design. The examples may also help us to focus on the styles used in design of computer artifacts, and they may themselves furnish some inspiration for design of computer artifacts.

Suggested readings: On Bauhaus see e.g. the complete collection of documents in (Stein 1969), and on Postmodern architecture e.g. (Jencks 1984). In (Thachara 1988) several articles argue that design of computer artifacts in the Postmodern era is becoming just as important a design field as architectural and industrial design. For several examples of the relevance of architectural reflections in our field see contributions in (Norman & Draper 1986), and for further references (Ehn 1988). (Norman 1988) is a nice introduction to stories about industrial design. On professional designers way of thinking and doing in other design fields, see especially (Schön 1983).

4.6 Philosophy of Design

Philosophy, especially theory of knowledge, and its relation to design is a topic for a curriculum in design of computer artifacts that is inherent in many of the other topics, but that also should have a place of its own. A topic like this can be argued for any art and science, but it is crucial to design of computer artifacts, since this subject matter, as conceived here, is both interdisciplinary and concerned with basic conditions for knowledge production in practice.

Furthermore, we need philosophy to reflect on two different kinds of theories—*operational* and *substantial* theories (Bunge 1967). Operational theory on how to do design is characteristic to the art and science of *design* as opposed to natural, social and human sciences in general. Such theories espouse design norms or rules to be followed and design artifacts to be used. But as in any other art or science

we also need substantial theory about the phenomenon of design, about e.g. what kind of social, historical, scientific, artistic, and technical activity design is. Especially, we need approaches that allow us to integrate operational and substantial theories.

Suggested readings: Churchman's 'interpretations' of philosophical ideas from Lebniz, Locke, Kant, Hegel and Singer in *Design of Inquiring Systems* is a great source to learn from (Churchman 1981). In a modest, but similar way, I have tried to 'interpret' philosophical ideas from Marx, Wittgenstein, and Heidegger to make sense to an art and science of designing computer artifacts (Ehn 1988). As Churchman I argue the need for a more fundamental understanding of design than the one offered by the dominating rationalistic systems thinking based on the Cartesian dualism of the objective and the subjective, of body and mind. This need concerns knowledge in the design process, as well as knowledge in doing design research, and knowledge in theories of design. The direction outlined for theory and practice is towards practical understanding of the games people play in design and use of computer artifacts. *Human practice and understanding in everyday life should be taken as the ontological and epistemological point of departure in inquiries into design and use of computer artifacts.*

However, the heritage from the rational Cartesian tradition in our field is overwhelming. Maybe the best way to contest this hegemony is by reading the two Wittgensteins—the young rational hero in *Tractatus Logico-Philosophiocus* (Wittgenstein 1923), and the mature philosopher in *Philosophical Investigations,* who is trying to show us how he at first got philosophy wrong (Wittgenstein 1953).

4.7 Practical Design

With the programme outlined it is obvious that a reduction of a curriculum in design of computer artifacts to what can be taught and learned as detached theoretical reflections is contradictory. Both philosophical investigations and examples from architecture and industrial design point in direction of the importance of practical understanding as knowledge by experience (Polanyi 1957), by familiarity (Wittgenstein 1953), and as reflection-in-action (Schön 1983) in design. Hence practical design taught by professional designers of computer artifacts and their use, both as experimentation in a master-apprentice relation and as investigations into real cases, seems most fundamental to our curriculum.

This practical education should include examples of, and experimentation with, both a wide variety of application domains, and a wide range of design artifacts and norms and rules for their use. Hence, application domains should not be restricted to traditional administrative systems, but include new domains e.g. computer support for graphic arts, or cooperative work in small groups. Neither should design artifacts and methods be restricted to more or less formal system description techniques. A new focus should be on design artifacts that support involvement and experience, like the use of prototypes and mock-ups,

exploratory programming environments, scenarios, maps, and even role playing.

The suggestions for practical design are certainly easier to proclaim than to implement. Not only do we lack economic resources for such a practical orientation. More serious is the fact that many of us teaching the art and science of designing computer artifacts probably lack the practical competence of professional design. That is perhaps the most threatening consequence of the proposal.

Suggested readings: For the importance of practice, and the master-apprentice relation in industrial design see e.g. (Mayall 1979). At the annual *Information systems Research seminar In Scandinavia* there have also been some important practical suggestions for teaching the art and science of designing computer artifacts (Greenbaum & Mathiassen 87) and (Øgrim 1988).

5 A Possible Transcendence?

I do not see the above listed topics of a curriculum in design of computer artifacts as a replacement of what is already taught in computer and information science. But any discipline that teaches and does research on the subject matter of design of computer artifacts is strongly encouraged to let it take its place by the side of what is already taught, because it covers some of the fundamental aspects of what design of computer artifacts is really about.

It has been my intention to provide arguments that any discipline which deals with the subject matter of design of computer artifacts should be able, theoretically and methodologically, to treat design as including aspects like interventions into practice, as communication between users and professional designers, as a creative process, and as a concerned human activity. And I have argued that any discipline dealing with such aspects of design will have to transcend a natural science foundation, regardless of whether the discipline is called computer science, information processing, informatics or information science.

The domain of the subject matter outlined is not primarily theory of physical events or logic inferences in a machine, but of concerned human activity within a background of tradition and conventions in designing and using these artifacts.

I find this subject matter truly interdisciplinary, with relations not only to aspects of traditional computer science, but also to theories and methods from social and human sciences such as sociology, psychology, anthropology, linguistics and business administration. Such relations have only marginally been touched upon in this paper, though the role of substantial theories and methods from these disciplines in a curriculum in design of computer artifacts is evident. Such relations and central topics like computer graphics, ergonomics, theories of organizations and of social change will have to be discussed in another paper.

Furthermore, have I not discussed research methods for a science of design of computer artifacts, another obvious topic for our curriculum. However I think that many research situations bear a family resemblance with the design approaches discussed here. What I have in mind is participatory actions research, explorative experimentation, and case studies—not as a replacement for detached

theoretical reflection in one or another theoretical context, but as their practical and empirical foundation.

Instead I have tried to focus on *a disciplinary base for the interdisciplinary subject matter of designing computer artifacts.* In doing this I have tried to benefit from developments in other more mature design disciplines, particularly using the theoretical and methodological discussions in architectural and industrial design as paradigm cases. However, recent developments in computer and information science have provided valuable contributions, as well.

There have also been indications that what professional designers really do is more art than science. This challenge to an art and science of designing computer artifacts particularly deserves further investigation. A first step could be to explore the traditional master-apprentice relation as one form for education in design of computer artifacts. A second is to consider the arts as paradigm cases for the design of computer artifacts.

However, the different contributions discussed in the outlined programme indicate that we already have an intellectually sound and teachable disciplinary base for an art and science of design of computer artifacts. Much still remains to be developed, especially when it comes to methods. And in many respects we have to understand "the design process [as] hiding behind the cloak of 'judgment' and 'experience"' (Simon 1969, p. 80), that Herbert Simon once started his rationalistic crusade against, and for which he has had so many followers. But this retreat to practical understanding is for theoretical reasons, not because of intellectual softness. Simon's rationalistic systems engineering approach may be an ever so elegant 'intellectually tough, analytic, partly formalizable, partly empirical, teachable doctrine about the design process" (Simon 1969, p. 58), but I see no reason for us to retreat to it as a paradigm case for an art and a science of design of computer artifacts.

As compared with the rationalistic approach the outlined program may be a step backwards on the road towards operational theory and methodology, but by being more fundamental, this may also allow us to take two steps forward in designing powerful computer artifacts to augment the skill of users rather than replacing them by artificial intelligence.

By taking this road, we will certainly also need to make extensive use of many of the theoretical and methodological findings in the natural science based research tradition in the design of computer artifacts. By understanding computers as the material we use in design of computer artifacts, the importance of traditional computer science knowledge to design is obvious. An architect that has no understanding of building materials and techniques may design aesthetic and socially useful houses, but no one will be able to live in them, if they cannot be physically constructed. The same holds for designers of computer artifacts.

The question is not whether the one kind of knowledge or the other is needed. Both the research-guiding interest in technical control, and the interest in inter-subjective communication are fundamental to an art and science of designing computer artifacts. The problem is that the interest in inter-subjective commu-

nication has—hitherto to a great extent—been neglected in research and education of our subject matter. There are good theoretical and practical reasons for changing this.

Does this transcendence require a new 1968 and another "student revolt"? Will the emancipatory knowledge interest and the ethics and aesthetics of systems development then be an incorporated part of our art and science of designing computer artifacts? In another twenty years we may know.

References

Abrahams, P., (1987). What is Computer Science. In *Communications of the ACM*, 30(6).

Ackoff, R.L., (1974). *Redesigning the Future.* John Wiley.

Alexander, C., (1964). *Notes on Synthesis of Form.* Havard University Press, Cambridge.

Alexander, C., (1984). The State of Art in Design Methods. In N. Cross, editor, *Developments in Design Methodology.* John Wiley & Sons Ltd, Bath.

Bansler, J., (1987). *Systemudvikling—teori og historie i skandinavisk perspektiv.* Studentlitteratur, Lund.

Broadbent, G., (1984). The Development of Design Methods. In N. Cross, editor, *Developments in Design Methodology.* John Wiley & Sons Ltd, Bath.

Bubenko, J., (1987). *Invitation to conference in Åre,* April 6 8.

Bunge, M., (1967). *Scientific Research. The Search for System. The Search for Truth.* Springer Verlag.

Bødker, S., P. Ehn, J. Knudsen, M. Kyng and K. Halskov-Madsen, (1988). Computer Support for Cooperative Design. In *Proceedings of CSCW '88*, Portland.

Checkland, P., (1981). *Systems Thinking, Systems Practice.* John Wiley and Sons, Chichester.

Churchman, C.W., (1971). *The Design of Inquiring Systems Basic concept of systems and organization.* Basic Books, New York .

Churchman, C.W., (1968). *The Systems Approach.* Delta, New York.

Cross, N., editor, (1984). *Developments in Design Methodology.* John Wiley & Sons Ltd, Bath.

Dreyfus, H. L. and S.D. Dreyfus, (1986). *Mind over Machine—the power of human intuition and expertise in the era of the computer,* Basil Blackwell, Glasgow.

Dreyfus, H. L., (1972). *What Computers Can't Do—A Critique of Artificial Reason.* Harper & Row, New York

Ehn, P., (1988). *Work-Oriented Design of Computer Artifacts.* Arbetslivscentrum, Falköping.

Ehn, P., (1989). Playing the Language Games of Design and Use. In *Proceedings of COIS '88*, Palo Alto, extended version in *Cypernetic*, (in print).

Floyd C., (1984). A Systematic Look at Prototyping in R. Budde *et al.*, editors, *Approaches to Prototyping.* Springer Verlag, Berlin.

Floyd, C., (1987). Outline of a Paradigm Change in Software Engineering. In G. Bjerknes *et al.*, editors, *Computers and Democracy—A Scandinavian Challenge.* Avebury, Aldershot.

Greenbaum. J. and L. Mathiassen, (1987). Zen and the Art of Teaching Systems Development. In P. J rvinen, editor, *The Report of the 10th IRIS Seminar*, University of Tampere, Tampere.

Habermas, J., (1968a). *Erkenntnis und Interesse.* Suhrkampf, Frankfurt.

Habermas, J., (1968b). *Technik und Wissenschaft als 'Ideologie'.* Suhrkampf, Frankfurt.
Kuhn, T.S., (1962). *The Structure of Scientific Revolution.* Chicago.
Kyng, M. (1988). Designing for a Dollar a Day. In *Proceedings of CSCW '88,* Portland.
Ivanov, K. (1972). *Quality Control of Information.* Department of Information Processing, Royal Institute of Technology, Stockholm.
Jencks, C., (1984). *The Language of Postmodern Architecture.* Rizzoli, New York.
Jones, J.C., (1970). *Design Methods—Seeds of Human Futures.* Wiley, New York.
Jones, J.C., (1984). How My Thoughts about Design Methods have Changed During the Years. In N. Cross, editor, *Developments in Design Methodology.* John Wiley & Sons Ltd, Bath.
Langefors, B., (1966). *Theoretical Analysis of Information Systems.* Studentlitteratur, Lund.
Lyytinen, K., (1986). *Information Systems Development as Social Action—Framework and Critical Implications.* Jyvaskyla Studies in Computer Science, Economics and Statistics, University of Jyvaskyla, Jyvaskyla.
Mathiassen, L., (1981). *Systemudvikling og Systemudviklingsmetode.* DAIMI PB-136, Department of Computer Science, University of Aarhus.
Mayall, W.H., (1979). *Principles in Design.* Design Council, London.
Naur, P. and B. Randell, editors, (1969). *Software Engineering.* Report from a conference sponsored by the NATO Science Committee. Brussels.
Naur, P., (1985). Intuition and Software Development. In *Formal Methods and Software Development,* Lecture Notes in Computer Science no 186, Springer Verlag.
Norman, D. and S. Draper, editors, (1986). *User Centered System Design.* Lawrence Erlbaum, London.
Norman, D.A., (1988). *The Psychology of Everyday Things.* Basic Books, New York.
Polanyi, M., (1957). *Personal Knowledge.* Routledge & Kegan Paul, London.
Rittel, H., (1984). Second-generation Design Methods. In N. Cross, editor, *Developments in Design Methodology.* John Wiley & Sons Ltd, Bath.
Schön D. A., (1983). *The Reflective Practitioner—How Professionals Think in Action.* Basic Books, New York.
Stein, J., editor, (1969). *Bauhaus.* MIT Press.
Simon, H., (1969). *The Sciences of the Artificial,* The MIT Press, Cambridge.
Suchman, L.A., (1987). *Plans and Situated Actions—The Problem of Human-Machine Communication.* Cambridge University Press, Wiltshire.
Thackara, J., editor, (1988). *Design After Modernism.* Thames and Hudson, New York.
Ulrich, W., (1980). The Metaphysics of Design: A Simon-Churchman "Debate". *Interfaces,* 10(2).
Winograd, T. and F. Flores, (1986). *Understanding Computers and Cognition—A new foundation for design,* Ablex, Norwood.
Wittgenstein, L., (1923). *Tractatus Logico-Philosophicus.* Kegan Paul.
Wittgenstein, L., (1953). *Philosophical Investigations.* Basil Blackwell, Oxford.
Øgrim, L., (1988). Project Work in System Development Education. In J. Kasbøll, editor, *Report of the 11th IRIS,* University of Oslo.