# Applying Design Methodology to Software Development

*Jonas Löwgren*
Department of Computer and Information Science
Linköping University, 581 83 Linköping, Sweden
Tel +46 13 281482 • Fax +46 13 142231 • Email jlo@ida.liu.se

**ABSTRACT**
*Professional software development, and specifically the external design of interactive systems, suffers from a tension between the normative development models being prescribed and the actual design work being performed. This tension manifests itself in, e.g., recurring problems with fluctuating requirements. I argue that this tension can be understood as the clash of two views on external design work: the engineering design and the creative design perspectives. To explain the tension and to lay a foundation for new ways to structure software development, I seek to apply critical insights and concepts from design methodology — the theoretical framework for creative design. The result is a development process in which external design is separated from internal design and construction. The external design work consists of conceptual, constitutive and consolidatory steps. The process shares some characteristics with participatory design, but the designer's expertise is recognized and identified.*

**KEYWORDS:** design methodology, professional software development, external design, creative design.

## 1. INTRODUCTION: TWO PERSPECTIVES ON DESIGN

The motivation for this paper is my increasing awareness, from research and professional experience in usability-oriented software design, that something is wrong with the common view of what software development is and ought to be.

My research group is focusing on usability-oriented systems development. Our goals are to improve professional practice in the direction of usability, and to contribute to the scientific body of knowledge by studying tools, techniques and methods in professional contexts using research methods from the social sciences. As a result of our studies, I have been fortunate to interact with a large number of professional software developers and to closely observe software development processes. I have also worked as a usability-oriented software designer in two consultancies. Many times, I have reflected on the remarkably bad fit between how the process is described in normative process models and methodologies, and how it is performed in practice.

As this paper tries to show, I attribute the misfit to different views of the external design work, i.e., design of the behavior and appearance of the system, the services it offers its users and its place in the organization. Briefly, I will argue that the normative models are typically based on an engineering design perspective whereas external design in many respects is better described from a creative design perspective.


DIS 95 Ann Arbor MI USA © 1995 ACM 0-89791-673-5/95/08..$3.50

### 1.1 Overview

The paper is structured in the following way: I start by painting a broad picture of the two perspectives of engineering design and creative design in order to highlight their differences. Section 2 argues that existing models of software development are based on the engineering design perspective, and introduces the research topic of design methodology which can be seen as descriptive and normative theoretical frameworks based on a creative design perspective. In section 3, I illustrate that software development in practice is better described in some respects as creative design. A conceivable conclusion of this observation is that it could be useful to apply design methodology theories to software development. In section 4, I draw on recent and central insights from design methodology to demonstrate how software development projects can be structured in a way that respects the creative nature of the external design work involved.

### 1.2 Engineering design and creative design

The purpose of the following descriptions is to paint a rather broad and over-generalized picture and to highlight the differences. I will use the terms *engineering design* and *creative design* to refer to perspectives or views rather than narrow and well-defined methodologies. Clearly, the picture I present is a caricature of professional practice and normative theory. Nevertheless, I find it quite useful as a vehicle for understanding and, hopefully, improve practice. The relations between the perspectives and more precise models will be addressed below.

Engineering design assumes that the "problem" to be solved is comprehensively and precisely described, preferably in the form of a requirement specification. The mission of engineering design is to find a solution to the problem. The solution must satisfice the requirements and other constraints, such as cost or performance. Engineering design work is amenable to structured descriptions and seen as a chain of transformations from the abstract (requirements) to the con-

| | Engineering design | Creative design |
|---|---|---|
| process as a whole | entirely convergent | has divergent aspects |
| key question | how: determining how to solve problem | why: is this the right problem to solve? |
| stages in the solution | sequential: one candidate solution is refined | parallel: many alternatives are explored |
| nature of process | analytical: can be described as structured | creative: inherently unpredictable |
| purpose | one satisficing solution | explore possibilities before committing |
| ownership | impersonal: designer is an objective instrument | personal: the designer is "present" in the design; assumes responsibility for social action |

**Table 1: A summary of generalized differences between the two perspectives.**

crete (resulting artifact). Moreover, this structurability makes engineering design impersonal.

In contrast, creative design is about understanding the problem as much as the resulting artifact. Creative design work is seen as a tight interplay between problem setting and problem solving. In this interplay, the design space is explored through the creation of many parallel ideas and concepts. The given assumptions regarding the problem are questioned on all levels. Creative design work is inherently unpredictable. Hence, the designer plays a personal role in the process.

Table 1 summarizes my view of the differences between the two perspectives on design work, organized around a number of key dimensions. It must be pointed out that the differences are overgeneralized in the table; again, the purpose is to illustrate two idealized perspectives.

## 2. THE TENSION BETWEEN MODELS AND PRACTICE

In this section, I argue that normative models of software development in professional practice in general are based on an engineering perspective. My claim is that there is a tension between such models and the working practice. I give an overview of the research area of design methodology, which can be seen as the theory of creative design, as a foundation for the proposed perspective shift in subsequent sections.

### 2.1 Models in professional software development

In the large development companies that we have been in contact with, company-wide normative models for software development are the rule rather than the exception. These models are typically based on software engineering approaches; regarding design, they typically emphasize the following.

- The separation of analysis and design.
- Design as a way to fulfil the requirement specification.
- Hierarchical decomposition of the design work.

The following quote, taken from a highly influential text on software engineering, is also illustrative:

> "The design process appears to be a process of adding formality as a design progresses with constant backtracking to correct earlier, less formal, designs. Thus, the designer starts with a very informal picture of the design and refines that by adding information and making the design more formal." (Sommerville, 1989, p. 178).

In short, professional software development models in general are based on an engineering design perspective. In one sense, this is perfectly appropriate given that they were intended to govern the *internal design*, i.e., the construction of the software. There is no doubt that they contribute to the verifiability, maintainability and other properties that are crucial to the quality of the product from a constructional point of view. The problem arises when they are interpreted as models for *external design*, i.e., design of the external behavior and appearance of the product, the services it offers its users and its place in the organization.

My claim is that there are tensions between the normative engineering design models and the actual work that is carried out in association with the external design. In section 3 below, I present a number of illustrations from various sources, highlighting the tensions. The remainder of the paper discusses opportunities for applying a creative design perspective to the external design work and what the implications of such a perspective shift would be. But first, it will be necessary to introduce a theoretical framework of creative design and the design disciplines: the field of research known as *design methodology*.

### 2.2 Design methodology

Design methodology is an interdisciplinary topic, attracting researchers from miscellaneous design disciplines such as architecture, engineering and industrial design. Roughly, it was consolidated as a scientific topic between 1962 and 1982. This period is conventionally divided into three generations (the following account is largely based on Cross, 1984; Lundequist, 1992).

The *first generation* was characterized by an engineering approach to design, in which the designer was pictured as an objective expert. Design work was described as an iterative problem-solving process with three broad phases: analysis, synthesis and evaluation. Design methods were described in a rationalistic manner; a famous example is *Design Methods*

by Jones (1970/1992) in which 35 methods are described and an input-output table is provided to aid the selection of method for a particular purpose.

The *second generation* represented a reaction against the simplified view of the design process, and particularly the notion of unbiased information gathering and structuring. User participation in design, and the consequent shift of the designer's role from expert to liberator of the users' needs and requirements, was explored. The view of design problems as ill-defined and the intertwining of problem setting and problem solving were given theoretical formulations. Design was seen as an interplay of basic concepts and modifying factors.

The *third generation* focused on the specific competence of the designer; design was seen as a distinctive kind of thinking, as fundamental to man's intellectual ability as, e.g., language. Philosophical and psychological concepts, such as tacit knowledge, pre-structuring and abduction, were employed in order to approach an appropriate epistemology for design methodology. In design practice, and specifically in architecture, the remark has been made that due to the emerging ideology of post-modernism, the architect has moved out of the design process to concentrate strictly on form and expression. The question has been put whether this represents aesthetic liberation of the architect or merely an escape from reality (Lund, 1990).

Lundequist (1992) summarizes the emerging topic of design methodology in four research themes:

- Normative attempts to formulate models for the design process and how it should be supported by tools and methods.
- Attempts to analyze what design problems consist of, how they should be solved and what makes them different from other types of problems.
- Descriptive attempts to document, analyze and report the nature of design work in practice.
- Philosophical reflections on design work and research, including the formulation of central concepts of design.

### 2.3 Analogies with HCI

To support the perspective shift that I will suggest below, it is interesting to compare the development of the design methodology field with the main directions in human-computer interaction (HCI) research, and specifically the parts of HCI that are oriented towards external design and the development of usable systems. The first generation of design methodology, characterized by engineering approaches to design, corresponds well with the great interest in usability engineering methodologies in the mid and late 80s. The focus in HCI at that time was on operational definitions of usability (see, e.g., Carroll and Rosson, 1985; Shackel, 1986; Whiteside et al., 1988, first part). Good et al. (1986, p. 241) summarize the objectives of usability engineering very distinctly:

> "Usability engineering is a process, grounded in classical engineering, which amounts to specifying, quantitatively and in advance, what characteristics and in what amounts the final product to be engineered is to have. The process is followed by actually building the product, and demonstrating that it does indeed have the planned-for characteristics. Engineering is not the process of building a perfect system with infinite resources. Rather, engineering is the process of economically building a working system that fulfills a need. Without measurable usability specifications, there is no way to determine the usability needs of a product, or to measure whether or not the finished product fulfills those needs. If we cannot measure usability, we cannot have a usability engineering."

In later years, the engineering approach to usability has been increasingly questioned in the HCI community. An illustrative example is the discussion by Whiteside and Wixon (1987) on the "dialectic nature" of the term usability engineering. They redefine usability as a purely subjective concept, determined by the extent to which software supports and enriches the ongoing experience of people who use it. This view clearly rules out all attempts to objectively quantify usability before building the system, and in fact the authors moved on to formulate an alternative development approach known as contextual design (see, e.g., Whiteside et al., 1988, second part, or Wixon et al., 1990).

Another aspect of the engineering approach that has been subjected to criticism is the tendency to assume that the use situations and needs can be known and specified before the system is built. Adler and Winograd (1992) state that design for usability must include design for coping with novelty, design for improvisation and design for adaptation. Moreover, they strongly emphasize the need to position design for usability in relation to organizational contexts, work design and equipment design.

Adler and Winograd (1992) also criticize the oversimplified view of the user as a psychological automaton without social context, which leads further on to another interesting trend in recent HCI: The move towards participatory design. In the last years, the HCI community has been embracing work on participatory design techniques originating in Scandinavian systems development research in the late 70s and early 80s (e.g., Muller, 1992; Kyng, 1994). The literature on the topic is growing rapidly and a special conference (PDC) has been held since 1990.

To conclude the analogy, it would appear that the part of HCI concerned with the development of usable systems has closely replicated the patterns of the first and second generation in design methodology.

## 3. ILLUSTRATING THE TENSION BETWEEN MODELS AND PRACTICE

This section contains a number of "vignettes" from different sources. Some are proper research data, gathered using scientific inquiry methods. Others are my personal impressions and recollections. The purpose of the illustrations is not to provide an axiomatic basis for deductive conclusions, but rather to paint a picture of external design work as it appears in professional software development practice.

### 3.1 The Delta project

The Delta project was initiated in 1992 as a collaboration between our research group and a large Swedish consultancy

in the sector of technical systems. Our main goals were to study the uptake and adaptation of usability-oriented methods in professional contexts, and to investigate the effects of collaboration between system developers and technical communicators.

The officially approved development model of the consultancy fits the above characterization of an engineering model. It comprises study and analysis phases, followed by top-down specification, design and implementation. The testing starts with modules and proceeds up to function tests of the joint system relative to the function specification.

Our joint work started with the development of a method extension, comprising usability engineering techniques and procedures for the collaboration between system developers and technical communicators. To assess the results, they were used in a customer project in the consultancy. We studied the project using participant-observation techniques. Data were collected in the form of field notes, project documents, design sketches and complementary interviews. Carlshamre (1994) provides a thorough presentation of the project, the data and conclusions regarding the research goals mentioned above.

The reason for bringing the Delta project up here is that the data provide numerous examples of external design work that was inconsistent with the "official" engineering design model.

The design group faced the task of designing one subsystem within a larger system, intended to support the maintenance of certain technical equipment. The specification that was given to the design group was brief and superficial. Consequentially, Carlshamre (1994) characterizes the early stages of the design work as "figuring out what to build." Studies of users and their work and discussions with the client were interspersed with attempts at designing the domain objects and services of the system. This behavior is clearly inconsistent with the engineering design perspective demanding separation between specification and design. The intertwining of problem setting and problem solving characteristic of creative design, on the other hand, fits the data well.

On numerous occasions, the designers found themselves generating multiple design alternatives on different levels of abstraction. This initially caused high-tempered and emotional debates, oriented towards determining the "best" alternative. Carlshamre (1994) describes the discussions as characterized by guesswork and a strong sense of individual ownership of the respective design ideas. The solution to the dilemma was found when a member of the design team proposed that tough decisions should be postponed until the usability tests, where the different alternatives could be tested. This resolvement is described as a relief in that no stakeholder would have to "surrender". A reasonable interpretation of the data is that the designers, trained in the engineering tradition, sought engineering design "solutions" to what is really a creative design phenomenon (parallel approach, personal ownership). An approach grounded in a creative design perspective would instead accept and thrive on the divergence of the design problems, encourage massively parallel design work and the generation of numerous alternatives in order to better understand the design problem at hand.

### 3.2 A story from personal experience

In the first half on 1994, I was employed as usability designer in a medium-sized consulting company. I was assigned to a large development project, involving some 25 system developers in two different cities. The system to be developed had been planned, specified and discussed with the customer for a year. A project plan had been prepared and it had been decided that the system would be strongly modularized. It would consist of a number of independent tools, working on a shared relational database. The project plan was extremely tight — everything would have to run exceptionally smoothly for us to meet the delivery deadline.

As I started identifying the work I was supposed to be doing, I found out that the user interface development activities mentioned in the project plan were expected to exclusively comprise programming of the user interface to the different modules. Neither user or context studies nor design or usability testing were planned. High-level usability coordination of the different modules was to be accomplished through the production of a project style guide.

I was explicitly forbidden to access the intended users, for political reasons that I was unable to penetrate. On the other hand, I had access to colleagues in the company who had spent many hours with the customers and the intended users.

The interesting aspect of this project, for the purpose of the current paper, is what happened as I started designing the user interface to one of the modules. I produced a first concept in the form of sketches on paper and a simple storyboard illustrating what I saw as core tasks. When I submitted it to the project manager (who had to underwrite all design decisions in the project, internal as well as external), he immediately approved of it and told me to start implementing it. To me, it was nothing but a first cut at articulating some design ideas and I did not feel that I was done. Instead of implementing the first concept, I produced three or four alternative designs, again using sketches and storyboards. At a meeting with the project manager and some of the other developers of the module, I presented the different concepts.

The reactions were most interesting: It was obvious to me that they had been very focused on finding one adequate "solution to the design problem." My first concept was originally seen as the answer they expected me to provide; moreover, they were not experienced and knowledgeable enough in user interface design to be able to assess it critically. Once I presented fundamentally different alternatives without indicating my preferences, they realized that there were many possible "solutions", and understood that the differences between the concepts reflected differences in ideas about the users' future work. A very lively and fruitful discussion followed, which resulted in (1) design being recognized by the project manager as an explorative activity, and (2) the articulation of a strategy for the product from the users' perspective. A notable long-term effect in the project was that the developers who had participated in the meeting started communicating about user-interface design by means of sketches and storyboards instead of written specifications.

To summarize this experience, the simple examples of creative design practice that I introduced were very well received

and understood. This caused the professional developers to reassess their own view of their work. My feeling was that I had provided something they wanted and needed.

### 3.3 Other empirical studies of software development

The research literature provides a few empirical studies of professional software development. On a broad level, four themes of relevance for my claim emerge from these studies. These themes are: models of the design process, early visions, the interweaving of problem setting and problem solving, and the role of the designer.

Rosson et al. (1988) describe professional software design as a three-stage process: information gathering, information processing and solution evaluation. Similarly, Malhotra et al. (1980) describe software design as three interdependent processes: goal elaboration, design generation and design evaluation. Nielsen and Aboulafia (1993) characterize the user interface design process as one of gradually evolving commitment, where intuition, imagination and unstructured analysis based in experience are the essential cognitive qualifications and the driving forces at design work. These descriptions are all consistent with similar empirical results in design methodology.

An important finding by Stolterman (1991) is that the first vision of the artifact to be designed "presents itself" to the system developers very early in the project. The analysis of users and their work, which according to the engineering design methods is intended to be an objective problem specification without prejudice, is instead seen as a way to verify or modify the first vision. Harker (1991) presents results indicating the danger of "design drift", by which she refers to the danger of losing the original understanding of the vision and goals if the process is prolonged or involves a large number of people. The "early vision" phenomenon has been long recognized in design methodology and given various theoretical formulations.

Numerous empirical studies (e.g., Hammond et al., 1983; Curtis et al., 1988; Due and Jørgensen, 1991; Harker, 1991; Bansler and Bødker, 1993; Walz et al., 1993) point to the emerging nature of the requirements and the understanding of the problem to be addressed by the design work. The phenomenon is sometimes framed as problems with formally specifying all requirements before designing the solution, and sometimes observed in the frequent iteration of prototypes and revisions. The design-methodological notion of problem setting interwoven with problem solving appears to cover all the data.

Frese and Hesse (1993) studied system developers' perception of user participation and found that the efficiency of the development process, the changeability of the product, the quality of the product, time efficiency, etc were all reduced in projects with high user participation. High user participation generated more change requests and the innovativeness of the projects were seen as lower by the developers. Poltrock (1989) showed that one feasible strategy to facilitate innovation was to acquire a "super designer with authority, knowledge about users' needs, vision and talent". I interpret the data as reflecting a need to consider the proper roles of the designer and the users in the development process. Again, design methodology can provide suitable starting points.

## 4. APPLYING DESIGN METHODOLOGY TO SOFTWARE PROJECTS

Based on the indications I have presented above, it seems that design methodology — the theoretical framework of creative design — may indeed provide useful insights also for software development, and specifically for external design. The research literature offers a few examples, most of which presenting design techniques that are applicable on the level of an individual designer or a design team. For example, the concept of Design Rationale (see HCI, 1991 for a survey) or, more generally, argumentative design can be seen as a reflection of creative design ideas about explorative and parallel design. Smets et al. (1994) discuss how the use of form to express concepts in industrial design can be applied also to the design of visual user interfaces. The techniques discussed by Bennett and Karat (1994) to facilitate design meetings emphasize, among other things, the development and maintenance of a group vision. Hooper (1986) reflects on the design domain of user interfaces rather than specific techniques, comparing it to the domain of architecture.

In this paper, I want to address the high-level issue of what design methodology may teach us about the nature of design projects and how they can be organized and managed in a way that respects the nature of the work involved. One of the crucial insights from design methodology, and one that seems to hold equally true in software development, is that *design problems and solutions cannot be separated.*

> "First, recognise that the 'right' requirements are in principle unknowable by users, customers, or designers at the start. Devise the design process, and the formal agreements between designers and customers and users, to be sensitive to what is learnt by any of the parties as the design evolves." (Jones, 1984, p. 213)

Current work in design methodology is focusing on how design processes should be shaped to address this inherent problem. The notion of participatory design, which was heavily criticized at the end of the second generation on the grounds of disregarding the designer's competence (see, e.g., Broadbent, 1979), is being reconsidered in the new light of concepts such as "knowledge work" and "learning organizations" in the post-industrial society (Clipson and Kornbluh, 1993). Participatory design is seen as a natural expression of larger-scale organizational development in the customer organization. The customer no longer buys an object (a building, in the case of architecture) but pays for help to develop his enterprise.

The misuse of designer competence is addressed by giving new theoretical formulations to the design process. The following section describes one such conceptualization of the design process; I then discuss the implications of applying design methodology theory to software development.

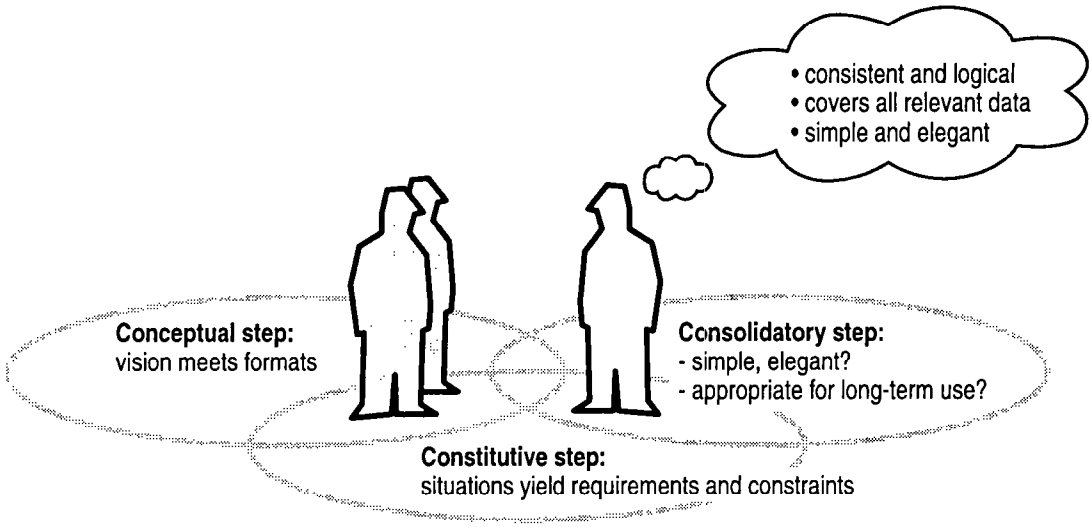


**Figure 1: The design process according to Lundequist and Ullmark. The designer (white) and the stakeholders (grey) stand in the middle of the three design steps. Note also that the designer is guided by his three ideal goals.**

### 4.1 A modern view of design from design methodology

The design methodologists and architects Lundequist and Ullmark (1993) have formulated a conceptualization of the design process (Figure 1), intended to facilitate the understanding of what happens in design and how designers use different kinds of knowledge. It consists of three qualitative steps: conceptual, constitutive and consolidatory. The three steps are not sequential in time, but rather represent different activities that the designer moves between in a rhythm determined by the design situation and the designer's individual characteristics.

The design process as a whole is guided by three goals or rules, that are ideal in the sense that they are desirable but certainly not always fulfilled in practice. The goals are the following.

- The designer's picture of the design must be consistent and logical, containing no internal contradictions.
- It must also cover all the relevant data at hand.
- If more than one design is possible, the simplest and most elegant one must be chosen.

The *conceptual* step is always guided by the designer's vision, which is typically not very distinct: It is unclear in parts and structure which makes it very hard to communicate to other people. The way to clarify the vision is to compare it with a number of known structures — which Lundequist and Ullmark call *formats* — in order to find one that corresponds adequately to the vision. The generation and test of formats is a very demanding task, since the number of possible formats is very large and the requirements in most cases are too complex to allow for systematic checking and analysis. Hence it is a task for the designer, based on an understanding of the users' needs and values. The designer searches for cases from his experience which in some way — metaphorically or otherwise — resembles the vision at hand.

The generation and test of formats is a physically tangible activity in which externalizations such as drawings and simple models are used. Different functions are symbolized in a simple way and put into the structure of the chosen format. This yields a set of rules for the next step of the work, which continues until a solution has been found or a contradiction appears. To deal with emerging contradictions, the designer can either modify the original format or abandon it in favor of another one.

In the conceptual step, the role of the users is primarily to help the designer get involved in the future use of the artifact. By sharing the same practice to a certain extent — through descriptions, explanations and demonstrations — different values and perspectives are articulated and confronted. As a result, the different stakeholders modify their original visions. It is important to notice, however, that this social process does not automatically lead to finding an appropriate format. On the contrary, the skill of the designer is crucial.

In the *constitutive* step, the design concept is confronted with typical situations which raises a number of questions about requirements and constraints. Some of these questions may have been known to the designer already in the conceptual step, but disregarded as being less important. Others appear as a result of increasing familiarity with the concept.

As the new requirements and constraints emerge, the designer tries to modify or extend the original concept which means that the format becomes harder to distinguish. Hence, the design work loses its direction and the designer does not know how to handle subsequent modifications and extensions. The time comes when the designer finds it necessary to go back to the conceptual step to rethink the whole concept, but on a higher level of knowledge. This second attempt to find an appropriate format is typically much more efficient and

precise than the first, due to the previous identification of a number of central features that can be used to guide the search process.

The users can play a more active role in the constitutive step than in the conceptual, now that a design concept is available for analysis and discussion. Of course, there may still be difficulties if the medium used to present the concept is not communicatively adequate, if the available time for participation is limited or if the participants are lacking in social skills.

The *consolidatory* step is concerned with refining the solution in terms of (1) simplicity and elegance, and (2) appropriateness for long-term use.

The first purpose of the consolidatory step has to do with aesthetic judgments, and certainly also with economical. Minimizing the resources necessary for realizing the design is an obvious goal in most practical settings.

The second purpose, i.e., the refinement of the design in the direction of long-term use, is focused on generality and flexibility. These issues are typically addressed already when the format is chosen in the conceptual step. Most common formats are based on solutions which have proven useful over a longer period of time in more than one specific functional context. However, it is always necessary to examine and reconstruct the solution as a final consolidation step.

Consolidation is a genuinely professional activity, and users who have been actively involved in the constitutive step sometimes feel sidestepped when the result of the consolidation is presented. In effect, "their" solution has been replaced by another one that seems strange and unfamiliar even though it is functionally acceptable.

### 4.2 Implications for software development

We have now seen that the external design of software is better described through the perspective of creative design than that of engineering design. We have also examined certain parts of the theory of creative design. In this section, I indicate what the implications are of applying creative design theory to the domain of external software design.

First, we must note that the design work in the design disciplines addressed by design methodology is separated from the construction of the designed artifact. The results of design may be prototypes, detailing the appearance and behavior of the final result, and specifications of requirements on the construction. The corresponding approach for software development is to separate external design from internal design and construction. This is also supported by Stolterman (1991), based on his empirical investigations of systems development and on a theoretically derived "ideal-oriented" design theory.

Secondly, participatory approaches to design appears to be the most promising way of addressing the inescapable intertwining of problem and solution. Lundequist and Ullmark's theory provides an important refinement to "traditional" participatory design approaches, namely that the constitutive step is the one where user participation is most crucial. The design concepts developed in the conceptual step serve as the common ground necessary for communication, provided that they are expressed in media that do not discriminate the users. Scandinavian research into participatory design has convincingly demonstrated the importance of equal-access media for prototyping (see, e.g., Bødker and Grønbæk, 1989; Ehn and Kyng, 1991; Madsen and Aiken, 1993).

Thirdly, the role of the designer as a professional, with expertise and judgment, is emphasized. This holds in particular for the conceptual and consolidatory steps. In the conceptual step, it takes a skilled designer to generate and test formats. The consolidatory step calls for aesthetic and economic judgment as well as experience in the long-term use effects of the situation at hand.

Historically, the designer has been seen as everything from the solitary genius of truly creative and inventive design ("Our job is to give the client, on time and on cost, not what he wants, but what he never dreamed he wanted; and when he gets it, he recognizes it as something he wanted all the time." Denys Lasdun, 1972, quoted in Cross, 1992) to the subservient facilitator or emancipatory ideologist, perhaps characteristic for some of the second generation design methodology work in the 70s. The role outlined above appears to offer an appealing compromise.

What, then, constitutes the expertise of the designer? Cross (1992) points to the abilities to deal with ill-defined problems by oscillating between problem setting and problem solving; to use early conjectures as a way to explore and define the design situation; to use strong constraints as generators of early solution concepts; to think by visual externalization; to recognize intuitive reasoning. These abilities were deduced from a survey of design methodology research and they do not appear incommensurable with our emerging view of external software design as a creative design process. Cohill (1991) has made a similar argument in favor of a creative design perspective on software development and describes the topical knowledge necessary for the software designer. He or she must have a foundation of design expertise, on which three areas of technical skill is added: information systems development, organizational behavior and ergonomics.

For the participatory design work in the constitutive step, Clipson and Kornbluh (1993) characterizes the role of the designer as a co-learner and knowledge sharer, an enabler of learning, a resource person who uses his expertise to enhance the developing work and learning competencies of the stakeholders, a convener of appropriate resources.

Finally, we must assess the view outlined above by considering the differences between external software design and other design disciplines. The most crucial difference is, of course, the material. Unlike architecture and other design disciplines, it is technically possible to evolve a software prototype into a final product. This may also help reduce the alienation in the consolidatory step, as the "distance" between the design concept and the final product is shorter than in, say, architecture. However, the principled case in favor of separating design and construction appears quite strong on theoretical and empirical grounds. We must also recognize, in reference to the alienation problem, that the knowledge employed by the designer to make judgments is still not available to the users even if the concept being judged is similar to a final product. Hence, I conclude that we should insist on the

separation between design and construction even though it is technically possible to dismiss it.

Another difference is the temporal nature of the artifact. Interactive systems are designed to have a certain behavior over time, whereas houses typically are not. The approach taken by Cohill (1991) is to reduce the difference by casting the designer as an "information architect" who is responsible for discovering information structures that enhance the users' intellectual capabilities. This is also consistent with recent trends in HCI (e.g., Olsen, 1992) to shift the focus of development efforts from user tasks to the information (or, to be more precise, the task domain objects) that the users need to access and manipulate in order to fulfil their goals.

## 5. SUMMARY

Software development informed by recent work in design methodology should aim at separating external design from internal design and construction. The external design work can consist of conceptual, constitutive and consolidatory steps where the designer's expertise is emphasized in the conceptual and consolidatory steps, and the constitutive step has a strong flavor of participatory design. This has implications for how we view the designer's role, as well as for the structure and management of software development projects.

A problem which remains to be addressed is how to deal with constructional aspects affecting the use of the final product. Response times, reliability, maintenance, etc., all affect the users' experience of the product (see Gould, 1988, for a discussion). Yet they cannot be adequately addressed in the design work since they are determined by the final construction. A tentative answer is that the designer — much like the architect — is responsible for coordinating the construction work in such a way that it satisfies the design vision to the greatest extent possible.

## ACKNOWLEDGMENTS

I am deeply grateful to Peter Ullmark of the Royal Institute of Technology, Stockholm, for providing me with information on current issues in design methodology. My thanks also go to my colleagues Pär Carlshamre, Mikael Ericsson and Torbjörn Näslund and to the anonymous reviewers for their incisive comments on previous versions.

## REFERENCES

Adler, P., Winograd, T. (1992). The usability challenge. In Adler, P., Winograd, T. (eds.) *Usability: Turning Technologies into Tools*, pp. 3-14. New York: Oxford University Press.

Bansler, J., Bødker, K. (1993). A reappraisal of structured analysis: Design in an organizational context. *ACM Trans. Information Systems* 11(2):165-193.

Bennett, J., Karat, J. (1994). Facilitating effective HCI design meetings. In *Human Factors in Computing Systems (CHI '94 Proceedings)*, pp. 198-204. New York: ACM Press.

Broadbent, G. (1979). The development of design methods. *Design Methods and Theories* 13(1):41-45. Reprinted in (Cross, 1984).

Bødker, S., Grønbæk, K. (1989). Cooperative prototyping experiments — Users and designers envision a dental case record system. Research report DAIMI PB-292, Computer Science Department, Aarhus University, Denmark.

Carlshamre, P. (1994). A field study in usability engineering: Bringing in the technical communicators. Research report LiTH-IDA-R-94-44, Dept. of Computer and Information Science, Linköping University, Sweden.

Carroll, J., Rosson, M. (1985). Usability specifications as a tool in iterative development. In Hartson, H. R. (ed.) *Advances in Human-Computer Interaction*, pp. 1-28. Norwood, NJ: Ablex.

Clipson, C., Kornbluh, H. (1993). Designing and learning: Towards a redefinition of participatory workplace design. In Törnqvist, A., Ullmark, P. (eds.) *Appropriate architecture: Workplace design in post-industrial society*, pp. 35-44. IACTH 1993:1, Chalmers University of Technology, Sweden.

Cohill, A. (1991). Information architecture and the design process. In Karat, J. (ed.) *Taking Software Design Seriously*, pp. 95-113. Boston: Academic Press.

Cross, N. (1984). *Developments in design methodology*. Chichester: John Wiley.

Cross, N. (1992). Design ability. *Nordisk Arkitekturforskning* 1992:4, pp. 19-25.

Due, B., Jørgensen, A. (1991). User interface designers in action. In *Precedings of Human Jobs and Computer Interfaces Conf.*, pp. 274-282. University of Tampere, Finland.

Ehn, P., Kyng, M. (1991). Cardboard computers: Mocking-it-up or hands-on the future. In Greenbaum, J., Kyng, M. (eds.) *Design at Work: Cooperative design of computer systems*. Hillsdale, NJ: Lawrence Erlbaum.

Frese, M., Hesse, W. (1993). The work situation in software development — Results of an empirical study. *Software Engineering Notes* 18(3):65-72.

Good, M., Spine, T., Whiteside, J., George, P. (1986). User-derived impact analysis as a tool for usability engineering. In *Human Factors in Computing Systems (CHI '86 Proceedings)*, pp. 241-246. New York: ACM Press.

Gould, J. (1988). How to design usable systems. In Helander, M. (ed.) *Handbook of Human-Computer Interaction*, pp. 757-789. Amsterdam: Elsevier.

Hammond, N., Jørgensen, A., MacLean, A., Barnard, P., Long, J. (1983). Design practice and interface usability: Evidence from interviews with designers. In *Human Factors in Computing Systems (CHI '83 Proceedings)*, pp. 40-44. New York: ACM Press.

Harker, S. (1991). Requirements specification and the role of prototyping in current practice. In Karat, J. (ed.) *Taking Software Design Seriously*, pp. 339-354. Boston: Academic Press.

Hooper, K. (1986). Architectural design: An analogy. In Norman, D., Draper, S. (eds.) *User-Centered System Design*, pp. 9-23. Hillsdale, NJ: Lawrence Erlbaum.

Human-Computer Interaction (1991). Special issue on Design Rationale. Vol. 6, no. 3-4.

Jones, J. C. (1984). Continuous design and redesign. In Jones, J. C. *Essays in Design*. Chichester: John Wiley.

Jones, J. C. (1992). *Design methods. Second edition.* New York: Van Nostrand Reinhold. First published in 1970.

Kyng, M. (1994). Scandinavian design: Users in product development. In *Human Factors in Computing Systems (CHI '94 Proceedings)*, pp. 3-9. New York: ACM Press.

Lund, N.-O. (1990). Designprocessen — Metoder og teorier. [The design process — Methods and theories]. In *Hur bra hus kommer till: ArkitekturMuseet Årsbok 1990*, pp. 69-77. Stockholm, Sweden. In Danish.

Lundequist, J. (1992). Om designteorins uppkomst. [On the origins of Design Methodology]. *Nordisk Arkitekturforskning* 1992:4, pp. 7-18. In Swedish.

Lundequist, J., Ullmark, P. (1993). Conceptual, constituent and consolidatory phases — New concepts for the design of industrial buildings. In Törnqvist, A., Ullmark, P. (eds.) *Appropriate architecture: Workplace design in post-industrial society*, pp. 85-90. IACTH 1993:1, Chalmers University of Technology, Sweden.

Madsen, K., Aiken, P. (1993). Experiences using cooperative interactive storyboard prototyping. *Communications of the ACM* 36(4):57-64.

Malhotra, A., Thomas, J., Carroll, J., Miller, L. (1980). Cognitive processes in design. *Int. J. Man-Machine Studies* 12:119-140.

Muller, M. (1992). Retrospective on a year of participatory design using the Pictive technique. In *Human Factors in Computing Systems (CHI '92 Proceedings)*, pp. 455-462. New York: ACM Press.

Nielsen, J., Aboulafia, A. (1993). Designing the user interfaces — The role of intuition and imagination (1992). In *Human Factors in Computing Systems (INTERCHI '93 Adjunct Proceedings)*, pp. 209-210. New York: ACM Press.

Olsen, D. (1992). User interface architectures for an information age. In Monk, A., Diaper, D., Harrison, M. (eds.) *People and Computers VII*, pp. 29-41. Cambridge: Cambridge University Press.

Poltrock, S. (1989). Innovation in user interface development: Obstacles and opportunities. In *Human Factors in Computing Systems (CHI '89 Proceedings)*, pp. 191-195. New York: ACM Press.

Rosson, M. B., Maass, S., Kellogg, W. (1988). The designer as user: Building requirements for design tools from design practice. *Communications of the ACM* 31(11):1288-1297.

Shackel, B. (1986). Ergonomics in design for usability. In Harrison, M., Monk, A. (eds.) *People and Computers: Designing for Usability*, pp. 44-64. Cambridge: Cambridge University Press.

Smets, G., Overbeeke, K., Gaver, W. (1994). Form-giving: Expressing the nonobvious. In *Human Factors in Computing Systems (CHI '94 Proceedings)*, pp. 79-84. New York: ACM Press.

Sommerville, I. (1989). *Software engineering*. Wokingham, England: Addison-Wesley. Third edition.

Stolterman, E. (1991). Designarbetets dolda rationalitet — En studie av metodik och praktik inom systemutveckling. [The hidden rationality of design work — A study in the methodology and practice of systems development]. Dissertation UMADP-RRIPCS 14.91, Umeå University, Sweden. In Swedish.

Walz, D., Elam, J., Curtis, B. (1993). Inside a software design team: Knowledge acquisition, sharing and integration. *Communications of the ACM* 36(10):63-77.

Whiteside, J., Bennett, J., Holtzblatt, K. (1988). Usability engineering: Our experience and evolution. In Helander, M. (ed.) *Handbook of Human-Computer Interaction*, pp. 791-817. Amsterdam: Elsevier.

Whiteside, J., Wixon, D. (1987). The dialectics of usability engineering. In Bullinger, H.-J., Shackel, B. (eds.) *Human-Computer Interaction — Interact '87*, pp. 17-20. Amsterdam: North-Holland.

Wixon, D., Holzblatt, K., Knox, S. (1990). Contextual design: An emergent view of system design. In *Human Factors in Computing Systems (CHI '90 Proceedings)*, pp. 329-336. New York: ACM Press.