DESIGN AS PROBLEM HANDLING – OUTLINE OF A FRAMEWORK

Anders Ekholm
Design Methodology, Construction and Architecture, Lund Institute of Technology/Lund University
Anders.Ekholm@caad.lth.se

SUMMARY

This paper presents an ongoing work concerning the development of a theoretical framework for design. The framework is intended to be applied to the development of information systems for architectural design. The paper combines ontological and epistemological perspectives using Bunge's work as a starting point, supplemented with Schön's studies of the general nature of design, and with examples of architectural design from Janson's empirical studies of the Swedish architect Jan Gezelius' works, as well as the author's own reflections and conclusions.

INTRODUCTION

Background

Theory of Science is an important part of Philosophy and a mandatory part of most PhD-education. Theory of Design has never been the subject of similar interest. Increasingly though, the need for scientific inquiry into design has come into focus (Love 2002). Both science and design are problem handling activities that require trained skills to perform. The difference is that while science is directed towards how things are, design is directed towards how things ought to be (Simon 1981:132).

Design means to determine the properties of an artifact with regard to requirements from use, production and maintenance. Even the design process itself may have certain requirements and consequences for the end result. Thus, artifact design requires both substantial and methodological knowledge (Bunge 1983). Design may be conceived of both as a conceptual representation and as the process that results in the conceptual representation. The term 'design' has meaning both as noun and verb. The noun *design* means "a mental project or scheme in which means to an end are laid out", and the verb *design* means "to conceive and plan out in the mind" (Webster's 1999).

The knowledge needed in different fields of design is so diverse, e.g. concerning functions and material that it is impossible to learn to design apart from specific knowledge of the artifact to be designed Bunge (1985:227). The design of social systems, e.g. the organization of work groups in an enterprise, is different from the design of mechanical systems, e.g. the parts of a motor engine.

Although design requires specialized domain knowledge, questions for a "common ground", are raised from time to time, also recently (Friedman 2002). The present background is the increasing specialization and complexity of the design processes, the development of computer aided design and the need for a theoretically founded education in design at universities and institutes. A common theoretical framework would facilitate the development of discipline specific design methodologies, e.g. the development of object-oriented computer aided design tools for architectural design.

There are commonalities of both artifacts and design methodologies of different domains that would render such a framework possible. To start with, very generic theories about the nature of reality, including artifacts, are developed within the philosophical field of Ontology, among others Systems Theory, and common questions of cognition and knowledge, among others the concept of problem, are treated within the philosophical field of Epistemology. Semantics and Ethics are the other main fields of philosophy necessary in a framework for design, but they are not explicit subjects of this paper.

Technology, with its multiplicity of different disciplines, may in fact be defined as the science of artifact design. For example, the field of Architecture deals with architectural design of systems composed of both man and built environment. There is a need for a technological approach to architectural design since the discourse in architectural theory traditionally handles design questions from a philosophical

perspective (Nygaard 2002). A problem is that this discourse does not result in operational knowledge that can be utilized for development of design methodology and design tools.

The aim of this work is to develop an outline of a framework for design using a technological approach, and to reflect on consequences for the development of computer support for design of the built environment. It is carried out as a theoretical study and relates works within the fields of Ontology and Epistemology, especially Bunge's "Treatise on Basic Philosophy" (Bunge 1983) and Schön's "The Reflective Practitioner" (Schön 1983), with some of the author's own studies of the characteristics of the built environment seen as a complex socio-technical system with both material and cultural properties (Ekholm 1987a, 1987b, and 2002).

In the following sections are treated, firstly, artifacts and their properties, and secondly, design seen as problem handling. Thirdly, certain aspects of problem handling are discussed. In connection with each section, reflections are made on the application of the framework to the characteristics of architectural design. Reflections on requirements on IT-support for architectural design conclude the paper.

ONTOLOGY FOR DESIGN

Artifacts as systems

Concrete systems have composition, environment and structure (Bunge 1979). The composition is the set of parts of the system, the environment is the set of things that interact with the system, but are not regarded as part of it, and the structure is the set of relations, intrinsic and extrinsic. A concrete system has a state, i.e. all its properties at a certain point in time, and a history, i.e. its former states. Finally a system has laws, i.e. natural relations among its properties. Artifacts are man-made systems; they are tools that enable certain human activities and experiences. Besides having laws, artifacts also have rules, i.e. states prescribed in a social system.

Systems and their parts constitute a level order where things in lower levels are parts of systems in higher levels. In each level new properties emerge. The natural world is considered to consist of several levels of increasing complexity. The level order of atoms, molecules, cells, organs, individuals and social systems is an example. Levels orders also characterize the artificial world. For example, materials like clay, gravel and water, are parts of components like bricks and mortar that in their turn make up assemblies like masonry, which are part of buildings.

Properties of artifacts

Following Bunge, the properties of an artifact can be divided into material and cultural (Ekholm 2002). Material properties are regarded as independent of human experience; they are objective (Bunge 1983:80). Material properties are either mutual of a system and its environment, or intrinsic of a system. Mutual properties are based on either bonding or non-bonding relations. The three main categories of material property according to these distinctions are:

- *Functional* (mutual properties based on bonding relations to the environment), e.g. functions in relation to the environment, including side-effects and environmental effects.
- *Comparative* (mutual properties based on non-bonding relations to the environment), e.g. position, geometry, and temporal properties like moment of event, and rhythm.
- *Compositional* (composition and intrinsic properties), e.g. material, mass, density, surface structure and intrinsic processes.

Cultural properties are mutual properties of a subject and an object. Cultural properties are experiences and thoughts of a subject related to the artifact but not intended to represent its material properties. They may be *subjective* or *inter-subjective*, i.e. based on individual experience only or determined by convention. The three main categories of cultural property are:

- *Phenomenal* (mutual properties determined by the individual's experience), may be divided into *sensory*, e.g. color, loudness and brightness, and *introspective*, e.g. comfort, beauty and safety.

- *Symbolizing* (mutual properties based on interpretation of the system as a sign or symbol). They are information in a communication system. The symbolizing properties can be divided into linguistic or non-linguistic, books and road signs respectively have symbolizing properties.
- *Administrative* (mutual properties assigned to the system in an administrative context). Examples are ID, name, classification, and price but also descriptions and property declarations.

The objective of an artifact is to be efficient and acceptable for a given task. The sought for properties are expressed as *requirements* in a design brief. Statements about properties that the artifact may not have are called *restrictions*. Restrictions may be based on laws or on rules and are expressed as attributes or attribute values. For example, a restriction may express the maximum length of a new building delimited by neighboring houses on each side along a city street, as well as a rule for the amount of built ground.

DESIGN AS PROBLEM HANDLING

Knowledge

Bunge distinguishes between three kinds of knowledge (1983:72):

- *Sensory-motor* – e.g. knowing how to throw a ball, or to ride a bicycle;
- *Perceptual* – e.g. seeing the size of a space, or identifying a false note in a song;
- *Conceptual* – e.g. knowing the composition of a building, or the rules of chess.

Sensory-motor or perceptual knowledge need not be corresponded by conceptual knowledge. Knowledge that has not yet been, or maybe cannot be, expressed as a proposition is called tacit (ibid:77). There is an extensive discourse about tacit knowledge, which will not be treated further here. A classic account has been given by Polanyi (1958), and a critical reflection by Rolf (1991).

Problems emerge and are defined in relation to a domain or field of knowledge. A *field of knowledge* consists of a combined material and conceptual framework: A *material framework* has three main components (Bunge 1983:90):

1. A *system, C, of persons* with special training, and strong information relations amongst them, and who initiate or continue a tradition of belief or inquiry;
2. A *society* capable of supporting and encouraging, or at least tolerating, C;
3. A *domain of objects* of inquiry.

And a *conceptual framework* has the following seven components (ibid):

4. A *general outlook* or *philosophical background* composed of (a) an ontology or view on the nature of things, (b) an epistemology or view on the nature of knowledge, and (c) a morality concerning the proper ways of acquiring, diffusing and utilizing knowledge;
5. A *formal background* of logical or mathematical theories taken for granted in the course of the inquiry;
6. A *specific background* of propositions and procedures, drawn from other epistemic fields;
7. A *problematic* consisting of problems concerning the nature, value or use of the investigated objects as well as problems regarding other components of the epistemic field;
8. A *fund of knowledge* that consists of a collection of propositions and procedures obtained by members of C at previous times;
9. *Aims* or *goals* of the problem solving activity of the C's;
10. All the *general and special methods* utilizable in the epistemic field.

Janson describes the competence of an experienced architect as ”a *repertoire* of everything that belongs to the professional skill” (Janson 1998:190). Such a repertoire may in fact be identical with the background knowledge or conceptual framework described by Bunge. According to Janson: ”The repertoire includes not only acquaintance with the inherited traditions and contemporary views of the trade, but also those new experiences and insights that the individual architect has gained through own practice – which influence the relation to tradition and the way to use it” (ibid).

Problem

Purposeful action, including inquisitive behavior, “curiosity”, and a desire to understand, is a driving force that leads to the discovery of knowledge gaps or problems. In its widest sense, a problem could be described as “the difference between what is known and what one wishes or needs to know” (Bunge 1983:235). A problem is a *conceptual object*, with three main constituents: presupposition(s), generator and solution (ibid: 242,271). *Presuppositions* belong to the background knowledge applied in the problem handling process, they belong to the *approach*, or *conceptual framework*, of the problem (ibid:259,90). A *generator* is an attribute representing an extrinsic property of the investigated or invented system. A *solution*, finally, is a description of the composition and internal structure of a system that has the required properties. A tentative solution is called *hypothesis*.

For example, the design of a building presupposes certain architectural background knowledge. Typical generators would represent the user’s experience of homeliness, efficiency, beauty and status. The generators in this example are phenomenal properties. The designer must transform these into material properties of the building, e.g. adaptability, spatial organization, and construction products, and propose a solution with the required characteristics.

Problems may be divided into cognitive and practical (Bunge 1983:244). A cognitive problem may concern both perceptual and conceptual knowledge, e.g. the ability to experience a piece of music and understand its composition. A practical problem concerns both sensory-motor, perceptual and conceptual knowledge. It requires skilful action to bring about the intended properties of an artifact, e.g. to perform music or design a building that generates the intended impressions.

Problem handling

Knowledge is obtained through learning, which physiologically is equivalent to the development of new properties of the nervous system (Bunge 1983:64). Learning can be identical with study, which means that the individual acquires existing knowledge. New knowledge is obtained through problem handling.

Problem handling processes may be characterized as routine or research (Bunge 1983:262). A *closed* problem can be unambiguously defined, and its solution is known in principle. It may be solved by a *routine* that consists in choosing a generic solution and specifying the values of its attributes. Routine problem solving is terminated when the problem is solved, e.g. when the equation is solved or the needle in the haystack is found. An *open* problem is ambiguous, it cannot be clearly defined, and its solution is unknown. It requires *research* since new kinds of things must be investigated or invented. Research problems are characterized both by solving problems and by finding new problems.

Views in problem handling

The generator and solution of a problem may be explained as different views on a system. A generator is an extrinsic view, and the solution is a compositional view. An extrinsic view, also called *top-down*, regards the system’s mutual properties: functional, comparative, phenomenal and symbolizing, and the intrinsic view, called *bottom-up*, regards its compositional properties, also named *technical solution*, see Figure 1. The compositional properties of a system are basic to its mutual properties. The mutual properties of a system also depend on the given environment. A shift of environment may result in a shift of mutual properties. During an inquiry it is assumed that there is a many-to-many relationship between a system’s mutual and compositional properties. Different composition and intrinsic structure may have the same mutual properties, and, vice versa, the same mutual properties may be generated by different technical solutions, e.g. the wall function may be realized by masonry or in-situ concrete.

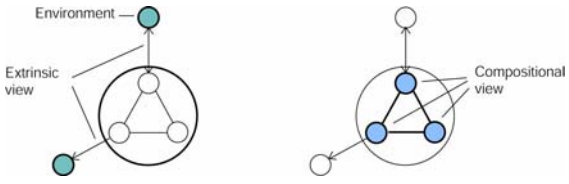

Figure 1 Extrinsic and compositional views on systems

In a systemic approach it is possible to distinguish between synthesis problems and analysis problems (Bunge 1983:274). A *synthesis problem* starts from requirements and restrictions of system and results in a hypothesis about the composition and intrinsic structure of the system. An *analysis problem* is the inverse of the synthesis problem; it starts from knowledge of composition and intrinsic structure, and results in an understanding of the mutual properties of a system in a given environment.

The problem handling cycle

Design is a problem handling process which intends to solve practical problems. Design may concern both routine and research problems. The result of the design process is a description of an artifact, including the knowledge gained in the process. Its aim is a satisfactory artifact, i.e. it must be both *efficient* and *acceptable* in its environment.

The design process starts by defining the problem, which includes determination of aim or purpose, choice of approach, compilation of relevant background knowledge, expressing generators as material properties of the artifact, and determining requirements and restrictions, see Figure 2. The problem definition is followed first by a *synthesis* and then by an *analysis*. Synthesis includes development of hypotheses and proposals for testing these. Analysis includes testing the properties of the solution relative a given environment, evaluating the results and supplementing the background knowledge with the solution knowledge. Analysis includes production of a concrete model, e.g. a drawing or a prototype that can tested. The new knowledge obtained through the sequence *problem definition – synthesis - analysis* is brought into the background knowledge. If the result does not solve the original problem, a new cycle could be initiated. Simon calls this problem handling sequence, which proceeds until a satisfactory solution is obtained, the ”Generator-Test Cycle” (Simon 1981:149).

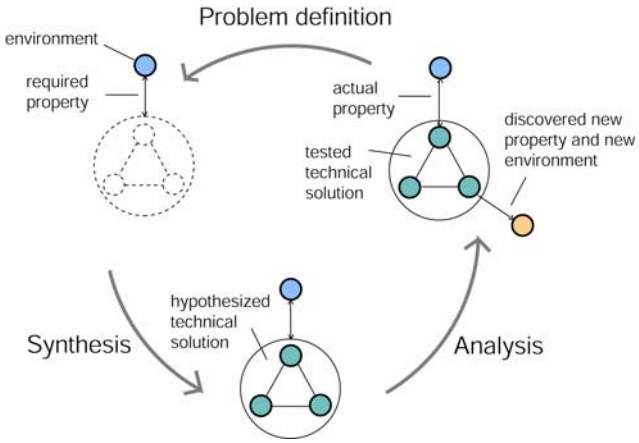

Figure 2 The cyclic course of problem handling

Janson’s account of the working practice of the Swedish architect Jan Gezelius has lead to observations that to a considerable extent are in agreement with the ”Generator-Test” model. Janson concludes that problem handling in design follows the sequence *driving conception – suggestion - exploration*. Janson describes the process as accumulating, neither cyclic nor linear but rather spiral shaped (Janson 1998:187).

Problem definition

In a real situation problems are not presented as givens (Schön 1983:40). Practical problems are characterized by uncertainty, disorder and indefiniteness. During the problem handling process the problem crystallizes and its conceptual context is clarified. According to Schön, ”problem-setting is a process in which, interactively, we name the things to which we will attend and *frame* the context in which we will attend to them” (ibid). It is only in the process of framing the problem, of determining the “problem setting” (Bunge 1983:236), that we can both clarify the goal and the problem, and determine the required means to solve it.

Another term for ”problem setting” is ”approach” or “view”. Bunge reminds us about the difference between a technological and a humanistic approach. The former is similar to that of applied science,

but its basis includes technological knowledge and its aim is to *control* natural systems as well as to design artificial ones. A humanistic approach is founded on the knowledge of human culture and handles problems of intellectual and artistic nature, uses conceptual methods and aims at *understanding* its referents (Bunge 1983:258).

Problem definition in design includes describing the practical situation or activity where the artifact is used, and determining the requirements and restrictions. Furthermore the generators must be translated into material properties of the artifact. Different methods for this translation have been invented, e.g. the ”House of Quality” method by the Japanese car manufacture industry (Cross 1994).

Synthesis

In contrast to implicit or tacit presuppositions, a *hypothesis* in its widest sense is an explicit assumption, conjecture, or educated guess of a solution to a problem. A hypothesis is developed through synthesis, a creative mental act. A hypothesis does not “come out of thin air” but is based on available knowledge (Bunge 1983:288). Bunge describes five, perhaps not mutually exclusive, ways to develop hypotheses:

1. Induction from observed cases
2. Associations of facts in time or place
3. Similarities of facts, e.g. analogies
4. Application of general principles
5. Imagination and invention.

Schön stresses the importance of “seeing-as” in creativity (1983:183). The analogies or metaphors thus created function as new generators in the design process. Seeing-as includes to reframe the situation. It is equivalent to exchange of parts of, or possibly the whole of, the conceptual framework: method, purpose, existing knowledge, problematic, conceptual tools, as well as the whole underlying world view. Reframing presupposes openness, questioning, critical scrutiny and conceptual ”width”.

In a semantic sense, a change of view is the same as a change of the conceptual context while the referent, i.e. the object of interest, remains the same. Along with the change of context follows consequences for which attributes that may characterize the object, and for the possible values that existing attributes may take. Janson exemplifies with the architect Gezelius’ work on the design of an exhibition building in a sensitive natural setting. The concept that represents the experience of the relation between the building and the environment was at one time suddenly changed from ”in the natural environment cautiously placed building” to ”a small public building”. This conceptual shift enabled a new colour, ”blue”, a new position, ”free and open”, and a new fully symmetrical shape (Janson 1998).

Analysis

The efficiency and acceptability of the proposed artifact must be evaluated. This is tried out in model studies and simulation, through prototype testing, or application in practice. Also the requirements and restrictions of a problem may be evaluated. Some of these are determined by the designer, others are decided by the client or authorities. The designer may impose unconscious restrictions that may or may not be unveiled as the design process proceeds. Evaluation may also concern the approach of the problem and the generators, as well as the designer’s own working method. The result of a completed problem handling cycle is added to the original background knowledge.

Schön shows how a practitioner continuously evaluates the problem handling method through ”reflection-in-practice” and ”reflection-on-practice” (Schön 1983:62). The former happens continuously during the different phases of problem handling. The latter is a critical valuation of the problem handling method as a whole, and can be used for further development of the conceptual knowledge applied to practical cases. He calls this ”knowing-in-practice” which leads to an incrementally increasing skill and a reduced need for theoretical reflection.

During design, concrete models are produced to enable evaluation through sensing, perceiving and interpreting. The model is to a certain extent ”independent” of the conceptual context in which it was conceived. Through reflection on a concrete model the original conceptual context may be redefined, and the hypotheses revised and further developed. Concretization in the form of model building and

design actions enable a conceptual liberation. It is assumable that this liberation is furthered by ambiguity and low precision of a sketch. At the same time it depends on the fantasy of the beholder and the inclination to associate and reflect, and transform the frame of reference.

Characteristics of architectural design

The conventional understanding of architectural design is that it mainly deals with cultural aspects of the built environment. However, it may be argued that a design problem includes the whole system man-tool-environment and that the design of the built environment includes both buildings and human activities (Ekholm 1987a). An activity together with the part of the environment that is used and experienced may be called a *situation* or, as by Alexander, a *pattern* (Alexander et al 1977).

From a compositional point of view, situations consist of people and equipment situated in a built environment. Decisive for the identification and delimitation of a situation is on the one hand its material properties, but also the experience of those that act in the situation. The environment must have cultural properties that support the activity. They may be sensory properties like light and colour, introspective properties like comfort or beauty, or symbolizing properties with a meaning that allows a person to identify with the situation.

Different parts of the built environment are controlled by different social systems. In principle this is a territorial subdivision (Habraken 1998). The built environment is a complex socio-technical system with several relatively independent levels of control. Examples of levels are “town-planning”, “building”, and “building space”. Each level is to a certain extent autonomous, which allows freedom for decisions about built “objects” in each level. Design of built objects is generally accommodated to this level structure so that certain objects are determined in lower levels and other objects in higher levels only.

REFLECTIONS ON INFORMATION SYSTEMS FOR DESIGN

Development praxis

The software problems connected with creative problem handling has not yet been solved. The development of new schema classes is not an integrated part of the design cycle, but is the subject of separate software development. Fischer (1998) distinguishes three intertwined levels of software design:

- The conceptual framework level
- The domain level
- The individual artifact level.

Here, the conceptual framework level should have a domain-independent architecture for building evolvable complex software systems. The domain level instantiates a higher level generic conceptual framework, and in its turn, the individual artifact level instantiates a domain level framework or schema. Each level has its own stakeholders: professional software developers, domain workers and clients respectively.

Fischer identifies seeding, evolutionary growth, and re-seeding as three separate phases in the dynamics of systems development. The “seeding process” is driven by system software developers in co-operation with domain experts and includes prototype development. During the evolutionary growth phase, domain experts use the software in practice. New requirements may surface that make end user modifications, including software programming, “re-seeding”, necessary.

Similar procedures characterize the development of object-oriented CAD-systems for building design. There is a dialogue between generic software developers and application developers. Influential users are engaged in a dialogue with application developers regarding the software performance, and skilled users are to a certain extent able to develop new object classes and change value spaces of existing attributes. For example GDL-programming allows users of ArchiCAD to extend its capabilities.

Some requirements on object-oriented design software

Information systems for design could be divided into synthesis and analysis tools. Analysis tools support the designer during evaluation, e.g. through simulations. Synthesis tools for open problem handling are more difficult to develop. Ideally, they should enable development of artifact models which can evolve and change as the insight of the problem is deepened in the design process. Traditional drawing based software only carry information about the designed object in an indirect way. The information exclusively deals with graphical objects and the design remains in the mind of the designer. Object-oriented design software on the other hand manages information about the design directly. The object model is external to the designer and resides in the computer system.

Today's object-oriented building design software suffers from major limitations from a design perspective. They are based on a fixed conceptual schema with a limited selection of object classes. Design objects are instantiated from classes in the schema. The user may determine attribute values for a specific object but cannot add new classes or value spaces. This software mainly supports routine problem solving. Support for research or innovative problem solving would require a flexible conceptual schema, user determined attributes and value spaces, and possibilities for reclassification of designed objects (Ekholm 2001). However, this is not simple to achieve, and according to Fischer above there may be other ways to achieve similar results.

In principle, design software should allow the designer instantiate a design object without other attributes than ID, and then incrementally specify other attributes, e.g. from an object library. Object-oriented CAD-systems for architectural design should allow the designer to specify information not just about building parts and spaces, but also about user activities and situations and other objects that the designer may think is important (Ekholm 2001). It is also necessary to be able to distinguish different levels of the built environment and to determine properties of built objects in different levels.

REFERENCES

Alexander C., Ishikawa S. and Silverstein M. (1977) *A Pattern Language.* Oxford University Press.

Bunge M. (1985). *Life Science, Social Science and Technology*. Vol 7 of Treatise on Basic Philosophy. D. Reidel.

Bunge M. (1983). *Exploring the World.* Vol 5 of Treatise on Basic Philosophy. D. Reidel.

Bunge M. (1979). *A World of Systems.* Vol 4 of Treatise on Basic Philosophy. D. Reidel.

Cross N. (1994). *Engineering Design Methods: Strategies for Product Design*. London: Wiley&Son.

Ekholm A. (2002). *Principles for classification of properties of construction objects*. In: Agger K., Christiansson P. and Howard R. (2002) Distributing Knowledge in Building - CIB W78 Conference 2002, Aarhus School of Architecture.

Ekholm A. (2001). Information systems for architectural design – experiences from the BAS·CAAD and ACTIVITY projects. Nordisk Arkitekturforskning nr 3, 2001.

Ekholm A. (1987b) The System Man-Building. Proceedings of the 31st Annual Meeting of the International Society for General Systems Research, Budapest.

Ekholm A. (1987a). *Systemet människa-byggnadsverk. Ett ontologiskt perspektiv*. Statens råd för byggnadsforskning R22:1987.

Fischer G. (1998) *Seeding, Evolutionary Growth and Reseeding: Constructing, Capturing and Evolving Knowledge in Domain-Oriented Design Environments*, Automated Software Engineering Vol. 5, No.4, October 1998, pp. 447-464,

Friedman, K (2002). *Design Methods*. Archives of PHD-DESIGN@JISCMAIL.AC.UK http://www.jiscmail.ac.uk/ 20/9 2002. Accessed 2002-12-12.

Habraken J. (1998). *The Structure of the Ordinary.* Cambridge: MIT Press.

Janson U. (1998). *Vägen till verket: Studier i Jan Gezelius arbetsprocess*. Daidalos.

Love T. (2002) *Constructing a coherent cross-disciplinary body of theory about designing and designs: some philosophical issues.* Design Studies vol 23, pp 345-361.

Merriam-Webster (1999). Merriam-Webster's Collegiate Dictionary. Electronic Edition ver. 1.5.

Nygaard E. (2002). *Arkitekturteorin mellan manifest och vetenskap*. Nordisk Arkitekturforskning, nr 3.

Polanyi M. (1958) *Personal Knowledge*. Chicago: University of Chicago Press.

Rolf B. (1991). *Profession och tyst kunskap*. Lund: Nya Doxa.

Schön D. (1983). *The Reflective Practitioner. How Professionals Think in Action*. Basic Books.

Simon H. (1981). *The Sciences of the Artificial*. Cambridge: MIT Press.