KMi Technical Report

KMi–TR–83

Knowledge-intensive design support:
An enquiry into the role of ontologies and analogous thinking in the design support

(Literature Review)

Martin Dzbor

Knowledge Media Institute
The Open University

Supervised by Dr. Zdenek Zdrahal and Dr. John Domingue

October 1999

Table of Contents

1 INTRODUCTION 5

1.1 Motivation 5

1.2 Review structure 6

2 PROBLEM FORMALISATION IN REAL-WORLD PRACTICE 7

2.1 Iterative nature of problems in engineering design 7

2.2 Insightful nature of problems in engineering design 9

2.2.1 Creativity in engineering design 9

2.2.2 Insight vs. familiar interpretations 12

2.3 Solution of inventive problems 13

2.4 Comparison and evaluation 15

2.5 Implications for the problem formalisation support 17

3 KNOWLEDGE-BASED DESIGN SUPPORT 19

3.1 Related research 19

3.1.1 Review of design support philosophies 19

3.1.2 Current positions 21

3.2 Introduction to knowledge modelling 21

3.3 Model of knowledge-based design support 23

3.4 Description of function and structure 26

3.5 Design by analogy 27

3.6 Justification of design decisions 28

3.7 Summary 29

4 SUMMARY AND DISCUSSION 32

4.1 Scope of further research 32

4.2 Discussion on the nature of design 33

4.3 Discussion on the importance of knowledge models 33

4.4 Research questions 34

5 REFERENCES 36

APPENDIX A

Scenario of knowledge-intensive formalisation 38

1 Introduction

This literature review builds a base for the further research of knowledge-intensive approach for the design problem formalisation. Engineering design is in general a complex activity; therefore my research will focus on one specific part of the design process, namely the problem formalisation.

1.1 Motivation

As Smithers, Conkie *et al.* (1990) suggest a design task may occur when an agent determines to change the status of the surrounding world. The purpose of design is then to give a specification how these changes may be done and the process that leads towards such specification is called *design*. Brown and Birmingham (1997a) add that design is a peculiar human activity and the desire to recast the world to suit our purposes is one of the defining characteristics of being human. Simon (1973) understands it as an example of a complex, ill-structured problem to which a solution may not be found until significant effort to understand the structure of the problem has been made. It is thus possible to agree with MacCallum (1990) who claims that design must be viewed as an intellectual, knowledge-rich activity of an agent.

In engineering design several mutually related activities may be recognised: description of a problem, solution, evaluation etc. Some of the phases and activities that build a complex process of engineering design are shown in Figure 1. The issue with the traditional CAD tools is that they provide support mostly to the later phases of design process such as solution generation and encoding (MacCallum 1990). The first phase – problem formalisation is only little researched and supported. I suggest that knowledge models may support designers in such creative activity as problem formalisation in design. Designers involved in engineering design often draw upon their previous experience when tackling uncertain design problems.

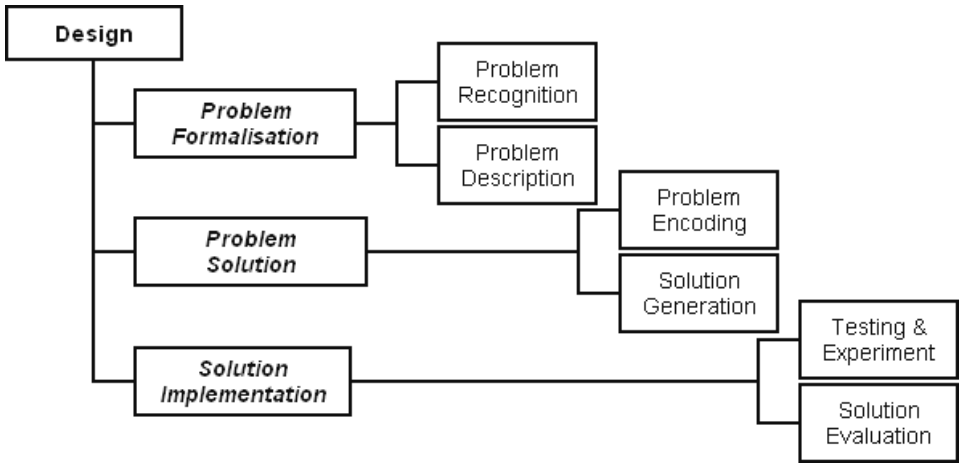

Figure 1. Engineering design activities

Several case studies of engineering design activities show that designers are often unable to formulate and explain their actions in details. Designers often rely on their intuition, insight, experience, etc. They may discover similarities between the current and previously solved problems without being aware of specific attributes that correspond. In other words, designers discover similarities not only on the level of a design solution, but also on a more abstract

level of requirement description. Several approaches in the engineering design community are reviewed in this paper that address, and describe the processes that can occur on such abstract level. The aim is to justify the claim that designers may benefit from receiving support for the design problem description.

In AI there is a strong group of researchers attempting to model human activities on *knowledge level* (introduced by Newell (1982)). The key assumption in knowledge-level modelling is that 'intelligent' behaviour of an agent could be explained in terms of agent's knowledge about the situation. In other words – there is a strong relationship between the knowledge and the behaviour of its possessor when tackling a problem. In the last decade various knowledge-level models were introduced for different domains. I suggest applying knowledge-level modelling in the designers' support for the problem formalisation in engineering design.

1.2 Review structure

In the following sections of the literature survey it is shown that the problem formalisation in design is an equally important activity as the problem solving. I will handle the problem formalisation as one phase within a more complex design process (Figure 1), and show that they are very closely related, indeed. Therefore it is worth to investigate the means how designers could be supported during the problem formalisation.

This review consists of 5 sections further divided into sub-sections. Section 1 contains an introduction and motivation for the research in progress. In Section 2 theoretical background will be discussed, and various approaches to engineering design will be presented. It is based on the research of Altshuller, Candy, Cross, Edmonds, Schön, and others. Several case studies of these researchers serve as examples to illustrate the existing issues in engineering design, as well as approaches to their solution. I conclude Section 2 with the comparison of reviewed theories, and identification of gaps for the future research.

Section 3 introduces a knowledge-based support to the design activities that were discussed in Section 2. In this section several related research works are reviewed, followed by discussions about some of the underlying theories. Separate sections are devoted to knowledge modelling, various support mechanisms in design, the role of analogy and different description languages in design, and finally the importance of justifications is emphasised. This section is based on research of Chandrasekaran *et al.*, Gero *et al.*, van Heijst *et al.*, Iwasaki *et al.* and others. The knowledge-intensive approach to problem formalisation will be the core of my further research.

Section 4 summarises the problem area that is reviewed in the paper in four separate sections: (i) implications for further research, (ii) discussion on the nature of design, (iii) discussion on the role of knowledge models and ontologies, and finally (iv) suggestions of further research and research questions. Some features of the knowledge-intensive approach are presented in Appendix A in form of scenario of design support. More details on the actual support to design activities will be published in future papers and reports; the work on them goes simultaneously with the review of the relevant research literature.

2 Problem formalisation in real-world practice

This section reviews current approaches and experience on problem formalisation process as described by different researchers in the engineering design community. The review is focused on how designers approach and tackle uncertain situations, how they translate the situations into soluble problems. The iterative nature of problem formalisation is highlighted in Section 2.1. Well-known case studies from Cross, and Candy and Edmonds on the creative design are reviewed in Section 2.2. Section 2.3 introduces briefly slightly different approach to the innovation and creativity in the design tasks based on the work of Altshuller and his followers. Finally in Section 2.4 the discussed approaches to creative engineering design are compared, associations between different theories are identified.

2.1 Iterative nature of problems in engineering design

It may seem straightforward to recognise and describe a design problem to be solved. I will show that it is indeed a knowledge- and experience-intensive activity. As Schön (1983) argues in his book: "*If it is true that professional practice has at least as much to do with finding the problem as with solving the problem found, it is also true that problem setting is a recognised professional activity.*" In this section some of the significant features of problems engineering design deals with are reviewed.

Schön observed in his research that in practice problems are not presented to designers with a detailed specification. Rather they must be built up from the *uncertain* situations. These situations require designers to perform a certain amount of work to convert them into a set of problems that can be described, solved and assessed according to formal criteria. Domain specific language is formal, and contains all concepts relevant to the situation at hand; which is in contrast to the 'language of customer' that is typically vague and ambiguous. What is more, the designer's language is more structure-oriented, whereas customer's is rather functionally oriented. Thus *problem formalisation* may be defined as a transformation of an initial uncertain situation to a set of formally soluble problems. I will use this term throughout the review, together with two activities that are part of problem formalisation – *recognition* and *description* of a problem in an uncertain situation.

When designers tackle an uncertain situation they usually begin with complex, ambiguous and ill-defined initial requirements. It is the complexity of the real-world situations that makes design a knowledge-intensive activity; and it is the ambiguity and uniqueness of initial requirements that makes design problems ill-defined. Schön in his theory of reflection-in-action claims that design must be inevitably interactive and cyclic process. He understands problem formalisation as a process, *in which designers interactively name objects TO which they will attend, and frame context IN which the will attend to the situation* (Schön 1983). Thus two separate but closely related activities are highlighted in the given definition – 'naming' of objects and 'framing' the context.

'Framing of a problem within a context' is concerned with the identification of the broader context of a problem within an uncertain situation at hand. It defines a viewpoint or a perspective to understand an uncertain situation. In other words, it may also translate customer's initial description into the language of a specific design domain. It is obvious that one situation may be described differently when different viewpoints are chosen. Viewpoint selection provides the starting point for the further work, and its main purpose is to narrow

down the vast space of possible design problem solutions. The further work of a designer consists of 'naming' the objects, relations and other features in the situation that are relevant to the current point of view. These named objects serve as a base for the formal problem description within the current viewing context. And since one situation can be described from various perspectives, in general the objects found and named using different frames will also be different.

Mulholland (1999) describes in his approach similar activities of professionals tackling uncertain situations. He builds his theory on the above mentioned theory of reflection-in-action and understands the entire process of problem solving (it includes also a phase of problem description) as iterative. It means that designer can change the context or a viewing perspective in the situation. This phase is depicted as a rectangle titled problem recognition in Figure 2. Such change may cause that similar situation tackled in the past will be recalled and new concepts to be identified and named in the current situation. Differences that may occur between previously solved problems and the current one may further lead to the possibility of applying another different perspective to understand and formally describe an uncertain situation. The procedure as described by Mulholland (1999) is depicted in Figure 2.

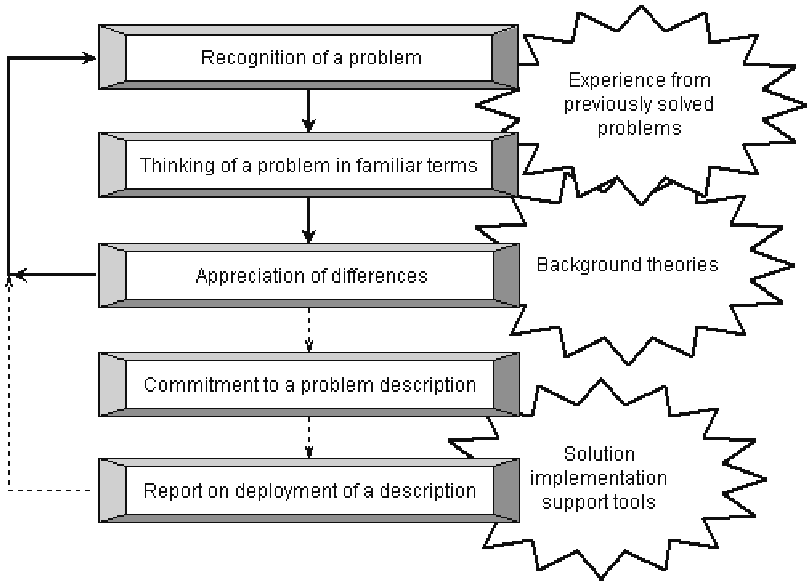

Figure 2. Flow model of problem solving using reflection

The first activity designers perform is problem recognition and description. It corresponds to an initial perspective taken on the uncertain situation at hand. The recognition and initial description may be based either on a background (domain) theory or designers' experience from similar situations in the past. Usually it will be based on a combination of both. Once designers have chosen an appropriate perspective, they can think of the problem in terms of the familiar situation experienced in the past. They can name the concepts within the current situation using terms from a similar situation. According to Schön and Mulholland this is the key activity. Two situations might be more or less similar, more or fewer differences might be found. A result or a difference that was originally not expected by designers can be perceived as 'a surprise'. If designers find something surprising, either positively or negatively in the problem description based on the chosen perspective reflection may occur. In other words, the surprising result is at least partially inconsistent with the domain theory or experience designers have drawn upon. Through the reflection on their activities and actions performed during a design process they can *make explicit the tacit understandings grown around the existing experience, and make a new sense of a unique situation* (Schön 1983).

Once designers appreciated differences between the current situation and the experienced one, they can commit to a solution strategy suggested by the previous situation. Otherwise, they may change their current perspective, and re-formulate the problem in a new way. The latter choice directly corresponds to the reflection on the problem formalisation. Also the former decision can later lead to the reflection, either on the solution strategy itself, or the problem recognition and description. Problem recognition, description, and solution thus significantly influence each other. The conclusions from Schön's research as presented above support the theory that problem is being solved simultaneously with the description of requirements set upon the final artefact. Nidamarthi, Chakrabarti *et al.* (1997) call this a co-evolution of requirements and solutions and also argues that it is an important feature of a design problem solving process.

Important features resulting from Schön's and Mulholand's findings in the design processes can be summarised in the following facets:

- Designers tackle problems in uncertain situations by drawing on familiar situations they have experienced in the past.
- If the familiar situation does not provide expected results, designers reflect on the current perspective and the problem description in order to identify the reasons of inconsistency.
- Designers work on their problems iteratively – they can modify the current perspective, which leads to a new description of a problem and/or solution. And vice-versa an existing problem description and/or solution may trigger the need to change the current perspective so that the new one explains the uncertain situation more closely.

2.2 Insightful nature of problems in engineering design

In contrast to research findings reviewed in the previous section Dominowski and Dallob (1995) warn that designers may often fail to solve the problems in uncertain situations due to fixation on the familiar and obvious interpretations of a situation. They understand familiar interpretations as a possible inhibitor of creativity in engineering design. Creativity is highlighted as a major professional strategy for tackling uncertain situations. Dominowski and Dallob claim that creativity plays a crucial role especially when situations to be tackled are ill defined, uncertain and ambiguous. They define problems as particular situations in which some goal is desired but the way to achieve it is unclear and must be discovered. In order to produce a new solution to a problem designers cannot be fixed to the obvious interpretations and descriptions; they must approach the problem using their intuition and insight to discover novel implications, relations and/or applications.

A large group of researchers in both engineering design and cognitive science communities is focused on the role creativity plays in problem understanding and solving in design. One of the examples is a theory of 'creative leaps' discussed by Nigel Cross in (Cross 1997). This work is reviewed in Section 2.2.1. Another study of creativity in engineering design is from Candy and Edmonds (1996), reviewed in Section 2.2.2. Both 'creativity' based and 'familiar situations' based approaches towards design are compared in Section 2.4.

2.2.1 Creativity in engineering design

Cross (1997) introduces the idea of 'creative leaps' based on the observed fact that many innovative design concepts emerge suddenly, almost in a form of 'enlightenment'. He defines the creative leap as *a sudden perception of a completely new perspective on the situation*. Designer's understanding of the situation at hand can be changed radically if a particular

concept discovered and named intuitively brings desired results. These results can cause a shift of an entire perspective taken to understand the problem in design – suddenly the designers are able to see the situation and the tackled problem differently. Suddenly they are able to explain certain features that they were unaware of before. In Cross's terminology designers have performed 'a creative leap' that has shifted them towards a new perspective.

In his paper Cross (1997) describes a case where designers work on a bicycle rack design. His designers have decided to implement the following design strategy:

1. explore the problem and write performance specifications;
2. generate a range of concepts;
3. evaluate and select the most promising concept;
4. develop concept and discuss it.

First they attempted to identify basic functions of the final artefact, and split the problem into several sub-problems. The initial problem recognition in conjunction with their knowledge of design processes and domain guided this work (step 1). When they attempted to solve the problem as they specified it in the beginning, they soon recognised new issues that had to be included into the problem description. They proposed a preliminary solution in the very early phases of the design process (step 2). However, all subsequent refinements of this initial approach did not solve some principal issues (e.g. water/mud spray thrown by a rear wheel).

The key breakpoint came when one of the designers started to look at the rack as a tray; and understood it as '*something like a bag is necessary*' (step 3). The newly identified concept of 'a tray' caused a shift in their perspective. This shift occurred in the third stage (evaluation) of their design strategy. They have reflected on the currently identified concepts ('a bag' and 'a panel') and existing issues (water/mud spray), and the reflection triggered their interest in the exploration of various concepts derived from the existing ones. Thus reflection and an insightful discovery of 'a tray' concept have modified their current understanding of an entire problem. Identified novel concepts have been further developed to fulfil other design requirements given at in the initial problem setting (step 4).

When trying to find justifications to describe designers' decisions, Cross (1997) discusses five models of a creative approach towards engineering design. A creative approach to the problem formalisation and solution can, according to Cross, occur by *combination, mutation, analogy, deployment of first principles,* or *emergence*. Designers often use these five design models in the early phases of design, especially in the problem formalisation phase. I will therefore describe them briefly in the following sub-sections.

2.2.1.1 Combination model

Probably the simplest case of a creative approach to both problem formalisation and solution is *combination*. It occurs by combining features from the existing problem descriptions or solutions into a new one. As Cross emphasises the new artefact need not be a completely novel concept. It may act as an innovative concept only in the frame of currently chosen perspective and current sub-set of goals. He illustrates this model by developing 'a tray' from a combination of 'a panel' and 'a bag'. It is clearly visible that the combination need not be structurally 'achievable' – i.e. we will not get a tray when we join a bag and a panel. The combination may occur on a more abstract level of the functionality of concepts.

2.2.1.2 Mutation model

The mutation model of creative design involves modifications to a particular feature of an existing solution (Cross 1997). In contrast to the combination model where particular features exist separately in different solutions and are only joined together, the mutated feature can give rise to a completely novel concept that did not occur before. Mutation can be triggered (similarly as a combination) by the insufficient performance of existing concepts, when designers are able to identify clearly the significant features to be mutated. Relevance of existing features to the expected functionality of an artefact can be used to assess the performance of existing and mutated concepts. An example might be modification of the feature 'height' of a concept 'tray' leading to a new concept of 'a box'.

2.2.1.3 Analogy model

Analogous thinking and reasoning is a well-researched discipline (Falkenhainer, Forbus *et al.* 1989; Gentner and Forbus 1991). It is also one of the models for the creative approach to design. Based on a higher-level abstraction and similarity between different situations, designers are able to apply terms from one situation in a (possibly) different one. Analogy is an often-used strategy for the understanding and formalising of a problem in a professional practice. As it was stated earlier, designers can think about the current problem, its formalisation and/or solution in terms of previously experienced and solved problems. In other words, they take an analogous approach to the current problem as they used to take for a similar problem in past. They may think analogously on various levels – e.g. formalise a problem using terms from a similar problem description to achieve a similar function or apply similar solving strategies to 'generate' a solution for a current problem etc.

Cross notes that designers must know how far to go with experimenting with the new combinations, mutations, or analogies. In general they may modify features of the existing concepts infinitely. However, in real-world practice they stop searching for new concepts, requirements and/or solutions when they are satisfied with the current one. It obviously does not dismiss an opportunity that someone else may find a new requirement or a better solution to the same problem in the future. The satisfaction is often purely subjective assessment of the situation and this is the reason why, according to Cross, such 'satisfaction' is difficult to realise in a computational support system.

2.2.1.4 First principles model

This model of creativity in the design is the least researched. It assumes the theoretical position that design proceeds from the given requirements or desired functions and based on them a designer seeks for an appropriate form or structure to achieve the desired functionality. The principles how to achieve specific functionality are known in advance. According to (Cross 1997) it is a form of an abductive reasoning process. In practice designers often create an initial model of a concept with features corresponding directly to the desired functions. Later they may further develop these features, assess them against the desired functionality of an artefact, and refine them further using other 'first principles' or other models of creative design.

2.2.1.5 Emergence model

Emergence occurs when analysing current perspective the existing properties are perceived as new. This corresponds to what Schön (1983) calls *reflective dialogue with the situation*. Applying a new perspective on a situation, a designer discovers interesting concepts already exist within the current description and/or solution of a problem, which s/he was unaware of. Thus emergence is very close to an 'accidental' or insightful approach to design.

2.2.2 Insight vs. familiar interpretations

Candy and Edmonds also attempt to describe creativity in design problem formalisation and solution. They propose that the key element in the creative approach is the designer's *act of insight by which certain focuses are chosen and combined in an innovative way*. In the following paragraphs I review their findings from the bicycle design case study (Candy and Edmonds 1996).

Candy and Edmonds understand each design situation as a unique case, which represents an extension to the knowledge, designers currently possess. Two mutually intertwined processes can occur in a work of a creative designer – first, the use of the existing knowledge to understand the current situation, and second the extension of this knowledge.

Several approaches to design problem solving – for instance, configuration design (Wielinga and Schreiber 1997; see also Brown and Birmingham 1997b) – emphasise the decomposition of a problem into a set of sub-problems, and solving these in a form of sub-solutions that lead to the final solution. On the other hand Candy and Edmonds put an emphasis on the *holistic approach*, where all individual features of the describing concepts and/or solutions must contribute to the main design goal. They claim that creative designers must always be aware of their ultimate goals, and it may be misleading to split the problem into sub-problems, tackle these separately, and still keep the awareness on the ultimate goal. This '*possession by the goal*' results in a deep immersion in the knowledge of the particular field. To achieve a freedom for the exploration designers often pursue their goals in parallel, at least in the early phases of design – compare to Section 2.2.1 where Cross's designers attempt to generate *a range of concepts* in the early phases of their work. Designers can use these parallel concepts later for the insightful development of novel concepts.

Parallelism in pursuing of the goal keeps designers open to the reflection on their actions. They are able to set up a problem in a certain way. Later when the performance of the current approach is unsatisfactory, they may change their initial formalisation; they may make a shift in their perspective. A similar practice can also lead to the generalisation and transfer of the experience from one item towards another one. The starting point in case of the bike designer in the reported case study (Candy and Edmonds 1996) were the existing traditions and models. And designer's *ability to make analogies with parts and products in other areas* has caused the extension of the traditional bicycle design space.

Candy and Edmonds highlight one specific feature of problem description process – that it is *a systematic expression* of a problem with all its inherent interdependencie*s*. The key term is 'systematic', i.e. approach the problem as a unique case that is being tackled in an established domain, with an established system of terms, and existing experience. Systematic approach enables designers to see the yet unanswered issues in the particular design domain.

For the classification purposes of creative actions in design Candy and Edmonds suggested five key elements participating and informing the creative problem formalisation and solution. The modified version of their scheme is shown in Figure 3. The key element in the diagram is a designer's expertise in the field (which includes the theoretical and empirical knowledge). The expertise influences among other aspects strategies and methods applicable for the formalisation and/or solution of a problem at hand. The difference between strategies and methods is conceptual and context dependent. Strategies are defined as the decisions taken in order to achieve goals, and methods as the sets of actions that carry out the chosen strategy. Candy and Edmonds identify basic strategy in their case as '*design → make → use (test) →*

iterate' (Candy and Edmonds 1996). The above-mentioned strategy makes explicit the *iterative nature* of processes in engineering design.

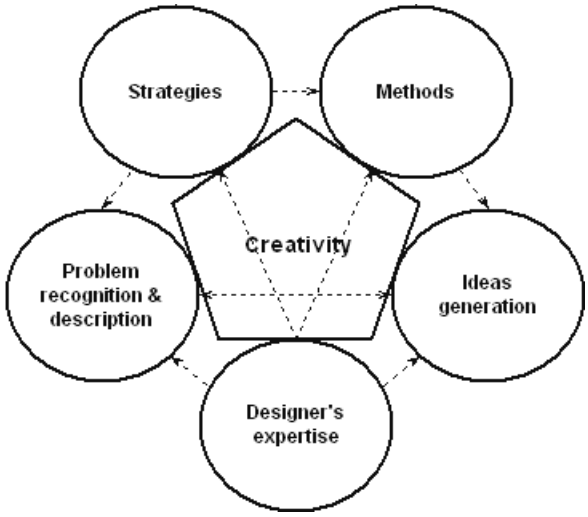

Figure 3. Elements of creative approach to problem setting and solving

A strategy drives the process of problem formalisation and/or solution by influencing two other elements of a creative approach – ideas generation, and problem recognition/description (see Figure 3). These two elements correspond basically to two steps of the proposed strategy – *make* (recognise a problem) and *design* (generate ideas). Problem formalised in this particular way can be then tested, and further refined. By refinement it is meant not only a sheer enhancement of the existing requirements on the final artefact or its features, but also a *re-formulation* of the problem. During re-formulation new requirements and/or concepts can be recognised and described. Both types of refinement are important because as an ultimate effect they may cause the perspective shift that is crucial for the invention of novel concepts.

Conclusions as presented by Candy and Edmonds emphasise the key role of insight in engineering design. However they do not agree with the claim from the beginning of Section 2.2 that familiar interpretation of a situation causes failures to solve problems. Contrary, they suggest that designers build on the obvious interpretations of an uncertain situation, and an insightful approach may extend their current knowledge and design space. Research work done by Cross, and Candy and Edmonds can be summarised in the following statements:

- The key element in engineering design is designers' insight, sudden, intuitive perception of an uncertain situation.
- Each design problem is a unique situation that can extend designers' current knowledge.
- Designers may combine existing approaches (familiar interpretations) to develop novel concepts. They also use analogies with products from other areas to perform innovative and creative design.

2.3 Solution of inventive problems

In this section I will discuss another approach to the solution of design problems, especially those having inventive character. Altshuller and his followers have been developing the theory as briefly described in the following parts over more than 40 years. It is based on the observation of thousands patents and inventions mostly from former Soviet Union (but includes also examples from other countries), and is today known as TRIZ[1] (Altshuller 1984).

[1] TRIZ stands for Russian equivalent of 'Theory of Solution of Inventive Problems'.

Altshuller's main argument was that *the evolution of all technical systems was governed by a limited set of objective laws* in contrast with blind search, trial and error methods. The analysis of patents revealed a classification of inventive solution into five distinguished categories. They range from simple improvements of a technical system requiring only existing knowledge (Level 1), through inventions that include a resolution of physical contradictions where knowledge from other areas might be necessary (Levels 2 and 3), to the developments of new technologies and phenomena (Levels 4 and 5). Altshuller concludes that majority of problems (77%) have inventive solutions that may be classified to two bottom levels, and argues that actual 'inventing' begins at level 3.

Basically the above mentioned classification corresponds roughly to that introduced by Qian and Gero (1996), where there exist three types/categories of design problems: (i) *routine*; (ii) *innovative*; and (iii) *creative*. They differ in how much knowledge is available in advance, how well structured is design knowledge, and what is to be found. Routine tasks tend to instantiate a fixed set of variables with values from pre-defined sets, whereas innovative and creative tasks modify and expand either one set, another one, or often both of them. However, it is quite impossible to distinguish where 'design' ends and 'invention' begins as Altshuller tried to do. As for instance, Watson and Perera (1997) point out creativity might be much more a social than a technical issue, and it is society that judges what is creative/inventive and what is not. In other words, we may perceive either design process that qualifies as creative or innovative, or it may be a final artefact deployed in a particular domain that is found to be innovative, creative or even inventive. Further discussion on the above issues of innovation and creativity in engineering design can be found e.g. in (Thompson and Lordan 1999).

In contrast with other research work in the area of design Altshuller builds his theory on a concept of *contradictions*. He claims that in every 'inventive situation' there are some contradictions present that must be removed by a solution. Contradictions may occur when trying to improve one characteristic of an artefact causes some other characteristic or parameter to deteriorate. A solution must thus be a compromise. Altshuller claims that a set of contradictory pairs of parameters is limited (actually very small in size) and it is possible to use it as an index to a library of so called *inventive principles*. Based on the patent analysis he has identified 40 basic principles and developed a table where specific contradictions might be removed by an application of one or more principles. The principles included, for instance, those of segmentation or merging of objects, preliminary actions, addition of a dimension, phase transition, preference of short-lived cheap objects and many more[2]. To solve an inventive design task several principles may be applied and combined. A sample implementation of a principle of segmentation may be the use of Venetian blinds instead of solid curtains or shades. The individual slats in Venetian blinds are connected together by a string so that it is possible to open or close them all at the same time (principle of merging).

The procedure of solving inventive (design) problems as proposed by Altshuller (1984) includes similar steps were already discussed in the earlier sections. It begins with a problem analysis and construction of a simplified problem model that reflects very narrow area where problems and contradictions occur. The next steps formulates an ideal solution that removes the problem from an area in focus, in this moment a vague problem has been described using only physical contradictions. The next two steps try to develop a solution to remove the identified contradictions. Here a reference to basic solving principles is often used.

[2] For a breakdown of inventive principles and examples see, for instance the following URL http://www.triz-journal.com/archives/97/jul/article2/fortyprinciples.html.

In case that an identified pair of contradictory parameters was not the right one and the task was not yet solved, the problem might be re-formulated, other contradictions may be put in focus, and/or other principles may be applied to achieve a satisfactory solution. The algorithm ends with the evaluation of the current solution, its effectiveness etc.

Altshuller's algorithm represents a typical example of a prescriptive approach to design. It is also a typical representative of the understanding creativity and innovation as features that belong to the design process and technique rather than design product and its deployment. Although we do not need to agree with such opinion and theory, the most significant achievement of Altshuller's work is the introduction of a limited set of (inventive) design principles that might be used in particular design situations. In this sense they may represent a base of 'best practices', heuristics that may help to find but do not guarantee the solution. They may serve as a framework for justification of certain design steps, to explain certain actions of designers, and help them to understand the relationship between requirements (possibly contradictory) and solutions (necessarily sound and consistent).

2.4 Comparison and evaluation

In this section I will compare the cases described in the previous sections. I will evaluate them with regard to the computational support for the problem formalisation task. Further I suggest possible relationships between the approach to engineering design based on the experience from familiar situations and reflection (Schön, Mulholland), the approach based on the insightful creativity (Cross, Candy & Edmonds), and finally the approach based on the algorithmic guidelines to support creativity (Altshuller). In Table 1 and Table 2 the steps of engineering design process and features characterising design are summarised, respectively.

Table 1. Steps of engineering design processes (Comparison)

	Schön	**Cross**	**Candy, Edmonds**	**Altshuller**
Step 1	*Recognise a problem*	*Explore the problem*	*Make*	*Analyse problem*
Step 2	*Name concepts*	*Write performance specification*		*Build problem model*
Step 3	*Frame problem in familiar context*	*Generate range of concepts*	*Design*	*Formulate contra-dictions*
Step 4	*Appreciate differences*	*Evaluate*	–	*Use table of contra-dictions and basic (inventive) principles*
Step 5	*Experiment in virtual worlds*	*Select the most promising concept*	*Use*	*Apply principles in the current problem*
Step 6	*Reflect on action*	*Develop concept using creative models*	*Iterate*	*If necessary, re-formulate problem*
Step 7	*Re-frame the problem*			

Table 1 identifies the corresponding steps within the different researchers' understanding of design process. The comparison is based on the functional similarity of design procedures recognised in their specific case studies. Table 1 shows possible mappings between the design steps as different researchers define them. From the observations Cross, and Candy & Edmonds did in their particular case studies, it is possible to conclude as shown in Table 1

that *the most promising concept selection and development* (in Cross's terms) is very similar activity to *the use and iteration* (in Candy and Edmonds's terms). Especially, the further development of concepts and iteration can be associated with Schön's and Mulholland's reflection on design actions. As it is visible, it is even possible to include Altshuller's work into pattern that has been built by previous researchers. Table 1 attempts to associate different understandings of design process. The research activities reviewed in the table do not have so much in common on the first glance; however, it is possible to find some pattern here.

Activities listed in Table 1 may be also compared to a well-known creative process model of Osborn and Parnes (in Thompson and Lordan 1999). This psychological model contains five stages, namely fact finding, problem finding, idea finding, solution finding and acceptance finding. They roughly correspond to the first five steps given in the table. To emphasise the iterative nature of engineering design tasks two more steps or stages were added – reflection and modification of the current understanding (re-formulation or re-framing).

Table 2. Corresponding features of different approaches to engineering design

	Schön	**Cross**	**Candy, Edmonds**	**Altshuller**
Features of a problem	*Ill-defined, unique, uncertain situation that needs a certain amount of work*	*Vague initial description that is evolving and changing*	*Unique case extending designer's current knowledge*	*Evolving technical system as unique case*
What triggers problem re-formulation	*'Surprising result'*	*'Creative leap', sudden understanding*	*'Act of insight'*	*Contradiction between parameters*
How can be problem re-formulated	*Shift of perspective, re-framing of a problem using different familiar situation*	*Combination, mutation, analogy, etc.*	*Analogies with parts & products in other areas*	*Using an inventive principle*

Table 2 compares the aspects that can cause the need for the problem re-structuring. Re-structuring is done by modification of the current problem description. Different approaches are given in the last row of Table 2. What triggers the re-structuring of the current understanding of a problem can be 'a surprising result' (Schön), a sudden perception of a new perspective – 'creative leap' (Cross), designer's immersion into the problem and an 'act of insight' (Candy and Edmonds), or an occurrence of a contradiction between requirements (Altshuller). Though these acts seem to be different, they all represent a sudden emergence of an unexpected issue, or an unsatisfactory behaviour that must be handled to bring the current state of the design problem in accordance with designer's goals. All researchers use the term '*designer's satisfaction*' when discussing the need for the problem re-structuring. Designer's satisfaction can be achieved in different ways. The shift in their current perspective of the problem understanding can be performed e.g. through combination, mutation or analogy. These models can cause the designers to focus on the novel concepts they were unaware of.

As we see from Table 2 Cross's 'creative leap' can be understood as a special instance of Schön's 'surprise' that occurred during the design process. The closest term in the work of Candy and Edmonds is the 'act of insight' by which designers can choose a certain focus on the uncertain situation. Altshuller's 'contradictions' can be transformed to the results that are inconsistent with designer's goals and thus may cause 'surprise'. All reviewed articles agree that a problem in a real-world practice is usually unique and cannot be described in full details

at the beginning of a designer's work. All researchers state explicitly that the problem description is evolving. The concept of the evolution of problem description – i.e. requirements set on the final artefact and solutions to them – is not new. Several researchers observed similar facts as mentioned earlier – requirements on the final artefact could not be set in advance. Often requirements emerge simultaneously as the solution process proceeds; many requirements are solution-specific, which means they may appear after the designer committed to a particular alternative solution. This feature of real-life design problems is known as *the co-evolution of requirements with solutions in design process* (Swartout and Balzer 1982; Nidamarthi, Chakrabarti *et al.* 1997)[3]. Such iterative approach fully corresponds and reflects designers' practice in the conditions of uncertainty and uniqueness when detailed description is not available at the beginning, but must be constructed 'on-the-fly'.

2.5 Implications for the problem formalisation support

I will use these findings to ground my hypothesis that designers can benefit from using support for problem formalisation. Lack of support for the problem formalisation in real-world practice results in an inefficient use of human and knowledge resources. Novice designers who come to the practice often need to '*re-invent the wheel*' (Fischer 1992) because they cannot draw upon as much previous experience. Providing them with support for the problem formalisation can result in saving their efforts in this area, which they can invest in their creativity. What is more important designers' performance is closely related to how they understand a problem at hand; what are they able to see and describe within a situation. Thus clarification in a structure of problem description might well help to tackle the problem, and entire uncertain situation. As it is reported in the previous sections, a lot of creative work in the problem formalisation is carried out using mostly unarticulated knowledge (often described as 'tacit' knowledge – term introduced by M. Polanyi, see e.g. in (Sveiby 1997). Therefore it is not possible to construct an 'automatic problem solver' as thought in the beginnings of AI. In contrast a requirement to *integrate the designer into the supportive computational system* emerged (Fischer 1992; Garrett 1998).

In the previous sections I attempted to justify the proposal that the problem formalisation is at least as important activity in tackling uncertain situations as the problem solving itself. Also Fischer (1992) points out that, to date engineers have been more concerned with the *transition from specification to implementation* rather than *with the problem of how faithfully specification really addressed problem to be solved*. He adds on that a *correct implementation of problem (...) is of little value if the specification is wrong to begin with*, and calls for more research and support for the problem formalisation.

The integration of a human and a computer is necessary because as stated earlier design situations are unique and uncertain, which implies that they cannot be tackled immediately from the beginning using formal methods. Laypersons agree with this point, however they will refer to designers' *artistry* (Schön 1983) to justify how it is possible to bridge the gap between the initial uncertainty and formally complete description of a problem. As we showed earlier, this 'artistry' depends heavily on designers' previous experience and domain knowledge that is available during design. The research activities described above suggest that skilled designers often organise their knowledge in such a way that they are able to find

[3] We can understand 'design requirements' as one of the forms of a problem description (e.g. they might be customer's descriptions of an uncertain situation, or ideas what is expected as a result of design process). A solution is then such instantiation of all design variables and parameters that satisfies given requirements.

out similarities and re-use previous experience. My research will focus on this particular fact – how to realise different structures in the design knowledge bases according which designers organise their knowledge. The role of knowledge bases (KB) in design is thus twofold:

1. KB serves as a source of knowledge for the problem description; a *vocabulary* of shared terms that may be used to communicate design problem description
2. KB serves as a *reference* vocabulary supporting retrieval of familiar/similar cases and their modification (knowledge re-use)

There were several attempts to construct various knowledge bases that would help to tackle design problems. Many approaches were essentially rule-based. Different approach to the construction of KB can be in the application of knowledge-level models for the representation of designers' theoretical and experiential knowledge from solving previous problems. As we show in the next section knowledge-level models enable designers to describe problem in 'different languages' and organise knowledge using different structures depending on the current phase of the design process. Fischer (1992) and Iwasaki, Fikes *et al.* (1993) point out that the language for the problem description in the early phases of design can differ from the language in the later stages – e.g. the implementation language. It may be a *language of doing* that *is familiar to all participants* (Fischer 1992); or a *functional description of a problem* that *enables to evaluate physical specification as it is developed* (Iwasaki, Fikes *et al.* 1993). On the contrary, language for solution specification will be more structurally oriented, will describe processes and behaviours occurring in the artefact and/or its components. The difference between two languages is not accidental; it reflects an evolving character of knowledge about a problem that is emphasised throughout the review.

3 Knowledge-based design support

Design systems for engineering have been developed basically in two parallel directions[4]:

1. The automated design systems; also called intelligent CAD (MacCallum 1990), whose aim was full or partial automation of the design process; the role of human designers was to give the initial requirements, evaluate solutions, build prototypes etc.
2. The design support systems that aimed at assisting human designers in their tasks either by recalling past cases (Watson and Perera 1997), critiquing and navigating (Fischer 1992), reasoning and consistency maintenance (Smithers, Conkie *et al.* 1990; Tang 1997), etc.

Engineering design was identified as a goal-oriented intellectual and knowledge-rich activity (MacCallum 1990; Smithers, Conkie *et al.* 1990). There were several attempts to develop a computer-based system supporting in one way or another designers and design process. For a further reference see e.g. (Smithers, Conkie *et al.* 1990; Tang 1997; Watson and Perera 1997). Knowledge-based design support system (KBDSS) may be basically defined as *a decision support system to enable designers to explore the structures of design problems and their solutions* by combining human design expertise with domain and design knowledge that might be stored in a system (Tang 1997). As seen from the definition, one of the aims of KBDSS is to assist designers in the exploration of structure of both design process and domain knowledge. KBDSS thus need means that would clarify the structure and simplify the exploration process. The next section gives an overview of related research activities aimed at the design support systems and identifies current positions of research in this area. In the remaining sections the application of knowledge models as a core technology for the knowledge base construction in a KBDSS is justified.

3.1 Related research

In this section different approaches towards design support will be reviewed, including their underlying architectures and philosophy. This review is rather informative than exhaustive; the aim is to address alternative research directions within the field of intelligent support to the engineering design.

3.1.1 Review of design support philosophies

The *Edinburgh Designer System* (EDS) is a knowledge-based design support system developed at the University of Edinburgh as a part of their 'AI in Design' Programme (Smithers, Conkie *et al.* 1990). EDS supports the work in multiple design contexts in mechanical engineering domain. EDS is based upon two basic pillars: so-called module class definitions and assumptions-based truth maintenance system (ATMS). Module class definition represents a generic model of a particular device (e.g. a turbine) that includes all descriptive variables, constants, parameters, constraints, etc. The design process control is performed through ATMS and initial designer's assumptions. EDS has a typical blackboard architecture where current design case contains different assumptions made by both – the designer in order to pursue particular direction in design, and EDS in order to initialise module class definitions and propagate them. EDS is able to utilise different knowledge sources to infer new datum as a consequence of given assumptions.

[4] Research as proposed here is to be included among the latter category of design support systems.

In EDS crucial decisions (assumptions) are made by human designers, EDS supports their work by propagating and inferring new knowledge. Human designers are also in control of the entire design process and are allowed to explore multiple alternative approaches. ATMS maintains contradictory assumptions in separate contexts that enable designers' exploration of a complex design space. Through assumptions EDS is able to address the issue of co-evolution of requirements and descriptions that together with a history of a particular design case construct so-called Design Description Document.

On similar principles is built a system called *Integrated Functional Modelling* (IFM) originating at the University of Cambridge. IFM is a further enhancement of the above mentioned architecture with some additional support tools (Tang 1997). Design knowledge base contains a hierarchy of components and relationships among them, sets of constraints and a detailed description of the components. The IFM starts with a formation of solution concepts that lead towards conceptual product model. Such model contains abstract information in terms of functionality and topology that satisfies given requirements. Conceptual model is further transformed into constrain-based model to search for a satisfactory embodiment solution. Search is performed through constraint satisfaction and propagation techniques.

IFM architecture thus combines rule-based and object-oriented knowledge representation into a system supporting work in multiple contexts and concurrent design environments. Designer's actions are treated as assumptions upon which system may perform some inferences.

Slightly different approach towards design support is visible in Goel et al.'s work on *KRITIK*, *IDEAL* and *E-KRITIK* systems, e.g. (Bhatta and Goel 1994; Prabhakar and Goel 1998). Goel et al. built their design support system on case-based reasoning principles and developed a specific scheme for design problem representation called Structure–Behaviour–Function (SBF) model. Both latter mentioned systems (supporting also 'across-domain' analogies) were constructed upon experiences from the former support system that was working with 'in-domain' analogies. For all systems a design case contains (i) functions that must be delivered by the product, (ii) structure of the product, and (iii) causal behavioural (SBF) model of the product. Similar cases are retrieved on a function-to-structure basis where functions serve as indexes in a knowledge base. Previous SBF models may be adapted for a new design case based on the functional similarity. IDEAL is also able to learn abstractions from several similar design cases attaining the same functionality.

The underlying technology of all mentioned systems are common ontologies developed by Bylander and Chandrasekaran (1985), functional modelling, case-based retrieval and adaptation of previous design cases. Especially IDEAL combines very efficiently case-based and model-based approaches.

Fischer and his team from the University of Colorado at Boulder pursued another alternative direction in design support. Their efforts were aimed at so-called Domain-Oriented Design Environments (DODE) in various design domains from user interfaces design to architectural layout planning (Fischer 1992; Nakakoji, Sumner *et al.* 1994). An underlying principle of most DODE-s was critiquing and argumentation. A generic architecture consisted of a catalogue of components and design cases, argumentative system, construction and simulation tools that were integrated through links satisfying various functional aims. Among them was a

catalogue explorer to browse and retrieve appropriate cases, an analyser as a critiquing system checking the compliance of current design with given guidelines and tactics, and an argumentation illustrator tool to help to understand critic's arguments.

DODE-s promote not only support to the design task, but also a deeper understanding of design, underlying principles, arguments and rationale. Domain knowledge is in the focus in this approach because as Fischer (1992) claims, the lack of domain orientation renders the amount of support that might provided and also decreases the shared understanding of problems among different parties involved.

3.1.2 Current positions

Several major streams of research activities exist in the development of knowledge-based systems for design. On one end of the scale are efforts to develop an *intelligent CAD* (MacCallum 1990) as a heuristic extension of traditional CAD. A large group of activities consists of *case-based design* systems that employ various CBR-related techniques to adapt previous solutions to suit current needs (Watson and Perera 1997). Another group of researchers pursues the usage of *prototypes* in design (Gero 1990), where a prototype is a model of design product or component to be adapted and instantiated for a specific case. When we move further towards more abstract end of the scale we arrive to re-usable *model-based design* systems (van Heijst, Schreiber *et al.* 1997), where design task and domain components are modelled in a hierarchical structure. Different models may be valid inside a single domain, across several domains, or for several tasks (Motta and Zdrahal 1998).

There were also several attempts to combine some of the above-mentioned principles and develop hybrid support systems. For instance: case-based and model-based support in IDEAL (Bhatta, Goel *et al.* 1994), rule-based and case-based approach in DODE-s (Fischer 1992), case-based and constraint satisfaction techniques of EDS and IFM (Smithers, Conkie *et al.* 1990; Tang 1997), etc. However, most of the current approaches address only some issues of design. They deal very well with case adaptation, while some of them pay less attention to other important issues in an intelligent support, such as learning, 'across-domain' knowledge transfer, and so on. These issues shall be addressed in further sections.

3.2 Introduction to knowledge modelling

As it was mentioned knowledge modelling can significantly improve explicit representation of knowledge about problems. In this section the broad term '*knowledge modelling*' will be discussed in more details. First, we define basic terms that are used throughout the remaining sections.

AI researchers attempted to develop the systems that can possess chunks of human expertise within a particular domain. Knowledge-based systems (or experts systems as they might be known before) were among the first publicly used results of AI research. These systems ought to mirror human ways of the problem solving in various domains. Researchers found it difficult to acquire necessary knowledge from the experts and prepare a good knowledge base. Therefore knowledge modelling became a popular research target for many research groups.

Newell (1982) introduced the 'knowledge level' assuming that an agent's actions during problem solving are determined by its knowledge about the problem. Thus it is possible to describe (or model) agent's actions on a level of knowledge it has about the problem at hand.

One approach to the construction of knowledge models[5] (KM) is based on entities known as common ontologies. Gruber (1993) understands ontology as *an explicit specification of conceptualisation; in AI systems, what 'exists' is what can be 'represented'*. Van Heijst, Schreiber *et al.* (1997) add it is a theory of *what can exist in the mind of knowledgeable agent*; and finally Chandrasekaran, Josephson *et al.* (1999) see ontology as *a representation vocabulary typically specialised to some domain or subject matter*. Ontologies may significantly clarify the structure of knowledge within a domain and provide means supporting knowledge communication, sharing, and re-use.

As it is visible from the definitions above, ontology serves as an abstract description of a knowledge representation for a particular domain. Ontology is used to describe problems but is not a part of the representation itself. An important aspect of the role of ontologies can be found in the definition given by van Heijst, Schreiber *et al.* (1997) that highlights the fact that ontologies may be used by different 'intelligent' agents, such as human experts, support systems etc. Since different agents use different representations, ontology often serves as a mediator in the interactions between them. When different agents want to communicate they need to agree on a shared vocabulary of terms (Chandrasekaran, Josephson *et al.* 1999).

Some researchers (Motta 1997) add a point that an inevitable feature that characterises any ontology is its *re-usability*. Though this is an important feature I think that re-usability can be understood as a special case of knowledge sharing. Knowledge sharing can occur either 'spatially' – i.e. different agents share the same ontology and use it as a mediator; or 'temporally' – i.e. the same agent can use knowledge in different times. The former type is usually referred to as knowledge sharing; the latter one as knowledge re-use. Therefore I did not include the condition of knowledge re-usability in the ontology definitions above.

For one problem domain several ontologies may be developed. They will differ mainly in their specificity and viewing perspective. In literature we can find examples of 'upper' (generic), 'middle' (prototypic), and 'lower' (problem specific) ontologies (Gero 1990; Chandrasekaran, Josephson *et al.* 1999). Upper ontologies are typically more abstract and are often suitable to more domains. On top of them are built middle and lower ontologies. Upper ontologies are often task oriented, whereas lower are more problem oriented (Motta and Zdrahal 1998). It is obvious that both types may have their specific roles in the design process. These roles will be reviewed and discussed with regard to the implications stated above. The first implication (see Section 2.5) demanded a KB as a source of domain and design knowledge. This implication is directly satisfied by definitions of KM and ontologies as representation vocabularies of subject specific terms that clarify and structure knowledge within KB. KM is a hierarchical structure itself, so it is possible to find an appropriate level of description depending on the design phase and the designer's experience[6].

The latter implication can be achieved through the hierarchical relationships between upper and lower ontologies. Similarity between previous design cases and current problem might be assessed in the network of hierarchically arranged KM of generic terms and terms specific for different domains. As an example we may have *Instrument*, *Agent* and *Object* as generic, task oriented terms of upper ontology. Then, *Missile*, *Launcher* and *Enemy-Plane* will be

[5] Although the full, correct term is 'knowledge-level models', it is usually possible to abbreviate it to 'knowledge models'; but we must keep in mind that we model an object or a problem on a knowledge level, not knowledge itself!

[6] KB is understood as a functional unit of KBDSS, whereas KM is a means how KB can be constructed. Often one term might be used instead the other; the meaning is given by context.

instantiations for the air defence KM; whereas *γ-Ray*, *γ-Generator* and *Tumour* will be instantiations of the same generic terms for the cancer treatment KM. Air defence and cancer treatment are 'parallel' KM that are related through a generic KM. Thus it is possible to see the above mentioned terms as equivalent because they address similar actions and instantiate the same generic terms.

As stated earlier ontology is a vocabulary of terms used for the problem description in a particular domain. It is obvious that several different types of ontologies can be constructed dependent on their purpose and focus. Van Heijst, Schreiber *et al.* (1997) differentiates the following four types of ontologies:

- *Generic ontologies* … their concepts are valid across different domains
- *Domain ontologies* … conceptualisation of a particular problem domain
- *Application ontologies* … particular task oriented extensions to domain ontologies
- *Representation ontologies* … framework to conceptualise the world

Brief explanations of different categories are listed above, and basically they correspond to 'upper', 'middle' and 'lower' levels in the hierarchy of ontologies. In my opinion the last category can be included among generic ontologies because its main purpose is to be used across more domains, to define the terms that describe other ontologies. Instead I would distinguish more explicitly between *factual knowledge of domain concepts* (what do we know and can use to express our knowledge) and *problem solving knowledge* (how do we express our knowledge). The knowledge about design processes, problem solving methods can be classified neither among generic nor domain ontologies. It may be partially domain-independent but not necessarily generic. Also it might be problem-specific but not necessarily expressed using application ontologies. Therefore some researchers understand it as a separate category (Chandrasekaran, Josephson *et al.* 1998; Motta and Zdrahal 1998).

Here it is important to emphasise the fact that the ontologies are not fixed, they can be modified and updated as designers extend their current knowledge. Such feature reflects a fact that not only problem descriptions (requirement and solutions) are evolving but also designers' empirical and theoretical knowledge represented by knowledge models is under constant evolution.

3.3 Model of knowledge-based design support

Suggested approach to knowledge-based design support is built mainly on the basic assumption in the 'design process' definition. We mentioned above that design is a *goal-oriented activity*. Design problem is being described with particular goals and expectations in mind. Typically design goals and designers' expectations are expressed in form of what functional requirements shall final system attain. On the other hand, designers' assumptions and description of designed system reflect what structures might be applied to achieve desired functionality. Thus, different 'languages' are available for designers to address different needs and phases of design process (Fischer 1992). It is quite difficult to separate them into independent design phases because, as we mentioned already, both languages are used almost simultaneously to express designers' current understanding of the problem.

The structure of 'design languages' plays an important role in the design process. As observed by (Smithers, Conkie *et al.* 1990; Candy and Edmonds 1996), carrying out a design task does not result only in a single solution, it also results in knowledge extension and better understanding of a certain type of design tasks. Thus knowledge acquired in tackling a

particular design problem is transferred to design and domain knowledge bases for future re-use. In this sense it is obvious that design process is understood as an *exploration task* rather than *search task* (Smithers, Conkie *et al.* 1990). Search is more or less unidirectional application of available knowledge to find a desired solution in some design space. Heuristic and domain specific knowledge might be used to improve search performance but this knowledge is rarely extended by a single design task. On the contrary when we assume design process as an exploration, from the very meaning of this term it is visible that new knowledge acquisition is 'expected' to extend the current resources.

As it was stated earlier, application of analogies and knowledge extension are basically two crucial features of a creative, professional design. Therefore any prescriptive model of design that is rigid and does not support knowledge transfer cannot be considered as a feasible *knowledge-based* design support system. Tang (1997) reviews various approaches to knowledge-based design systems and claims that most tools available cannot satisfactorily scale within and across domains, adapt to new contexts or acquire new knowledge. Partial remedy to these drawbacks lies in the case-based design (CBD) paradigm. Watson and Perera (1997) shows that CBD systems may facilitate not only analogous thinking and experience-based knowledge but also are able to learn by introducing new cases into knowledge repositories.

From what was mentioned so far, we might conclude that a viable KBDSS should be able to use all different categories of knowledge – i.e. theoretical and empirical knowledge, problem domain and design task oriented knowledge, as well as historical knowledge. The criteria satisfied by a successful KBDSS are abbreviated from (Tang 1997):

- Support to construction and extension of design knowledge bases;
- Support to solution derivation reflecting human actions in design processes;
- Support to explicit explanations and justifications of design decision steps;
- Support to multiple-context design, to the exploration of a problem from different perspectives;
- Support to knowledge transfer from individual design tasks to a shared knowledge base.

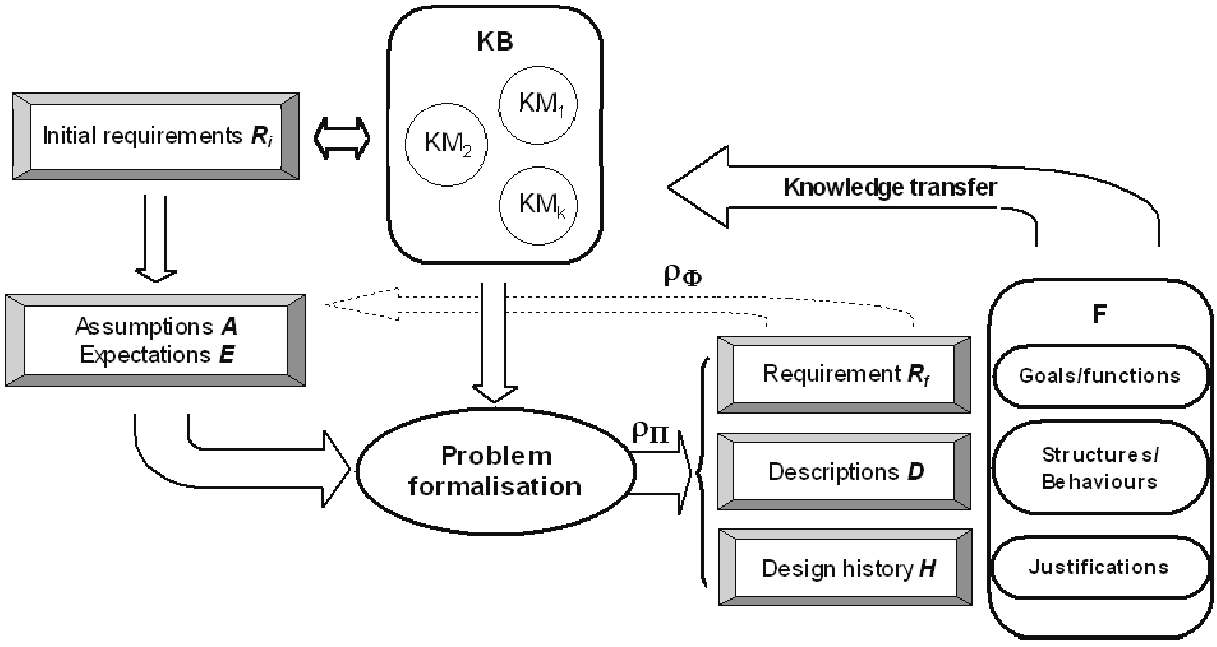

Figure 4. Generalised model of engineering design process

The following types of support have been identified in the works of van Heijst, Schreiber *et al.* (1997), and Motta and Zdrahal (1998):

- *Consistency & completeness checking*
- *Visualisation & reasoning process tracing*
- *Library of re-usable components & best practices*

By consistency checking a plausible combination or utilisation of the existing models can be assessed. Since designers use previously developed models as templates to guide the current problem description and provide the sets of familiar terms, it is important to use the terms in an appropriate way – i.e. consistently with an existing domain theory. As it was already mentioned designers use but are not limited to the existing theories, therefore the current theory can be changed in order to accommodate new observations. However, a consistency checker may bring designers' attention directly to the point that is inconsistent with the existing experience. In accordance with what was mentioned earlier, consistency checker may also support designers' reflection on their current problem.

Completeness checker has a similar role as the consistency checker. Its findings are useful for designers because they support the appreciation of suitability of the existing model for the current problem description. Support in form of a completeness checker can help to look for more analogies, and to explicate significant features of the current problem in order to re-use the previous model. This strategy can be specified as 'before developing a new model try to complete the chosen (similar) model'.

Visualisation and reasoning process tracing are very important modules in the support for problem formalisation. Through visualisation designers can perceive the knowledge-level description of the current and previous problems easier. Ontologies help to clarify the structure of the current knowledge about the problem at hand and support designers in the perception of the current problem in the context of previous experience – i.e. in the context of previously tackled problems. Through visualised reasoning process designers can appreciate the differences between the current and the existing problems faster.

Reasoning process tracing means that a supportive system should be able to provide the designer with the justification of performed or suggested action. It should be able to answer questions like '*Why problem XYZ was chosen as a similar problem?*' or '*How did this suggestion come out?*' More sophisticated knowledge-based systems can be able to tackle also '*Why not…*' and/or '*What if…*' questions.

Finally, a library of re-usable components can contain the models and best practices that were used to solve the design problems in the past. A library corresponds to what Schön calls *set of techniques to hand* to tackle an uncertain situation. Again, the re-usable components and/or best practices from the library may be either used directly in the new problem description, or more usually they will need to be adapted to the current problem. See e.g. Section 2.2.1 for models of creative design that can be applied for the adaptation of a library component for the current problem. A library can be in general organised according to different criteria, e.g. the domains of inquiry, the tasks to be solved, the methods to be implemented, the types of contradictions among parameters etc. (Altshuller 1984; van Heijst, Schreiber *et al.* 1997; Motta and Zdrahal 1998).

The summary of the section on knowledge modelling and application of ontologies for the design support may be written in the following facets:

- Designers can apply the familiar terms from previous problems *organised and structured in a form of ontology* to describe the current problem.
- Existing ontologies are applicable as *generic models/templates* to build on a top of.
- Ontologies are applicable as *a reference* in different kinds of design support, to justify or refute designers' decisions.
- Ontologies support *reasoning process* in the design support system.

3.4 Description of function and structure

Designers in a real-world practice tackle uncertain and unique situations. Uncertainty and uniqueness cause difficulties in the description of elements describing the current situation. Creative designers even try to avoid the low-level structural descriptions unless they understood higher-level features of the problem. In design they are usually concerned with terms like functions and behaviours of an artefact in contrast to its structure. Chandrasekaran (1993) developed a theory of relationship between *function, behaviour,* and *structure* (FBS) that was based on the causality of processes occurring in engineering design. Description of the problem according to the theory can be performed on three different 'levels':

a) description of an *intended function* of a device;
b) description of *how* the function *can be achieved* ('behaviour' of a device);
c) description of device *structure,* i.e. components, and their connections.

The FBS theory follows the strategy that recognition and description of functionality of a device is easier than that of device structure. Thus the description of a function precedes the structural description. Theory further suggests that a function can be observed without any commitment to particular structures. A *function* represents an effect the designed artefact shall achieve. The effect is a result of a causal process how the artefact and its elements interact and eventually achieve certain functionality. This process of causal interactions is known as *behaviour* of an artefact (or component). It can be explained (and later realised) either directly by a *structure* of an artefact, or through causal processes and relations that exist among structural components of an artefact.

Both functional and behavioural descriptions can be understood as higher-level perceptions of a design situation. Once designers are able to decide the function the final artefact shall achieve, it is possible to search for similar situations that delivered similar functions in the previous design situations. Thus a function might serve as a concept to assess the similarity between different situations (Qian and Gero 1996).

Also behavioural descriptions can help to find similar problems. Behaviour can imply a preliminary structure to a solution of a problem. Or it can introduce the causal processes that explain how the particular function is to be achieved. Often the behavioural description refers to the domain theories designers are familiar with. And even more important role of a behavioural description is that it can be used to describe the *thinking process* designers performed to describe the situation at hand. In general, one function can be achieved through different processes ('behaviours') and it is obvious that the same behaviour can be caused by many different structural configurations. The relationship between a function and structures is '*one to many*' (Qian and Gero 1996). This is also a reason why a problem in a real-world practice has usually more than a single solution.

Having the above-mentioned higher levels of problem description makes it possible for designers to re-use the previous experience that was encoded in a form of functional and

behavioural descriptions of previously tackled situations. When we return to the co-evolving requirements and solutions (see also Section 2.4) we may replace requirements with functional descriptions and solutions with structural ones. The co-evolution in this sense means that a modification in an intended function may change current components, and vice-versa, a modification to a structural component may bring designers' attention to functions they were unaware of in the previous step.

Though Chandrasekaran (1993) in his early work does not explicitly mention the iterative character of function, behaviour and structure descriptions, in his later research with Iwasaki, Fikes *et al.* (1993) this important feature of design problems is clearly pushed into fore-ground. In their paper Iwasaki, Fikes *et al.* (1993) illustrate the evolution of requirements (functions) and solutions (components) on a design of an electric power supply for a satellite. Their 'designers' begin with the fact that the device should generate power that is supplied to a load. With this requirement they identify their ultimate goal. Now they can move further to the question *how to achieve power generation*. In a library of components they found that se-veral different structural components deliver this functionality (e.g. battery, solar panel etc.).

First, they choose *a battery* but it is not suitable because after a certain time it can run out of power and the function of a device will not be achievable anymore. Similarly they find out that solar panel depends on the presence of sunlight. Finally they decide to combine these two approaches to the power generation. During the design process they have refined their initial problem perception from a simple *power generator* to *a power generator that works if sunlight is present or not, is able to accumulate energy, and is able to work for X years*. After several refining steps they found out that they needed a sensor/controller unit to prevent damaging of a device due to high temperature. This has led to further iteration in the problem refinement. Eventually the design was stopped when designers did not find other functional requirements that could be added, and the proposed solution satisfied all given requirements. However, the authors conclude that it does not mean that a better solution could not be found in the future (for instance, new technologies or materials might be available and included in the domain knowledge base).

3.5 Design by analogy

In the earlier sections I stated that *thinking in familiar terms* in order to describe a problem is typically realised by searching for analogies between previous and current problems. Qian and Gero (1996) suggest that *two designs are analogous if they have a similar function or similar behaviour; they may or may not have similar structures*. They further propose that the central idea of design by analogy is to capture the earlier mentioned one-to-many relation between function and structure in order to find different alternative descriptions and/or solutions. In other words – designers are often looking for different function ↔ structure pairs and discover how a particular function can be realised using different structure of components or what other functions can be realised through a particular structure.

To describe any particular design solution Qian and Gero (1996) exploited three basic descriptive elements– functions, behaviours and structures – and combined them into so called FBS paths. Term 'path' is taken from the graph theory because the relationship between functions, behaviours and structures can be easily depicted as a directed graph (tree). The point in the FBS path that can unambiguously identify a particular design process is a function of a designed device or component. Therefore Gero *et al.* suggest that function of the current problem must *match* the function of an analogous one if we want to design by

analogy. If the functions of the two problems match, the associated mappings for behaviours and structures may be found.

From a designer's point of view it is desirable to find analogies and mappings on the structural level. Gero *et al.* distinguish four types of structural components that a designer is able to describe – a) *element*; b) *attribute* of an element; c) *relation* between elements; and d) causal *process*. First three components are directly data concepts; the last one is a special concept to describe interactions among different elements. Two problem descriptions can be mapped through any of the listed concepts or their combinations.

An issue arising from the FBS-based approach towards analogy discovery is the need to describe the function of a designed artefact so that it is possible to look for analogies. Since functional description often serves also as reference for assessment of any candidate solutions it must be done using formal and unambiguous terms. However, designers when they begin to tackle new, uncertain situations do not have the vocabulary of such terms. Here they may take advantage of using the existing hierarchy of ontologies. Concepts contained in ontologies will serve as a vocabulary of functions known within a particular domain. Designers will thus describe the requirements on the current problem using the concepts in ontologies to find the right position of the current problem within available knowledge base.

3.6 Justification of design decisions

One of the assessment criteria of a feasible knowledge-based design support system was the existence of means for explanation and justification of performed actions. This kind of knowledge is very important because it increases the flexibility and usability of a knowledge base. Basically every decision step performed by a designer may be justified by one of the following references:

1. A reference to a domain theory base where the particular function or behaviour is defined. This may be, for instance, a reference to a physical law or a formula to calculate some parameter.
2. A reference to an existing problem description where the same decision step was performed and proved to be successful. This justification is based on an empirical result of one or several similar situations in the past.
3. A reference to a generalised principle or practice that typically proves as successful in the given situation. This may include various heuristics and rules that are applicable in the current stage. Here we may include for instance, many of Altshuller's principles of inventive design (see details in Section 2.3).

For several reasons it might be useful to record design decisions and their justifications. Justified decisions could be used later when a problem description is to be re-used or assessed. They may play significant role in the explanation of a reasoning process that leads to the selection of a particular alternative. And finally, designers may use the recorded justifications when they re-engineer an existing artefact. Justifications help to clarify the design process that might have been performed by different designers a long time ago.

Chandrasekaran, Goel *et al.* (1993) suggested that for certain types of problems functional description could record causal processes designer performed in the design of an artefact. As it was described in Section 3.5, function-oriented description of a problem is done through so called Function–Behaviour–Structure (FBS) paths. When designers decide to use a particular structural component in the problem description, an FBS path that is encoded in form of

knowledge model may justify this decision. In other words, designers could use the existing knowledge models to refer to: "*I will use this component in my description because in combination with component XYZ it behaves in a way that guarantees the achievement of the desired goal or function. The evidence is given in the model XYZ.*"

On the other hand the records can be also used to check the feasibility of the current problem description. The same model can be interpreted as: "*A particular component C_X was selected because it guarantees the achievement of a particular function. When I remove the component C_X, will be the desired functionality still achievable? I must be aware that the removal of this component must be followed by a selection of other component achieving the same goal, or I have to revise my goals or desired functionality in the current problem description.*"

And exactly this is proposed by Chandrasekaran, Goel *et al.* (1993) – to use functional description of a device (a problem) to check the feasibility of the solution in every single stage. All decisions designers make can be assessed against identified functions and existing domain knowledge. The proposed requirement or solution is feasible if it does not violate the achievement of the desired function, is consistent with an existing domain theory, or may be justified by a newly developed and proved theory/experience/rule. In other words it is feasible if at least one knowledge model can be found that contains a similar requirement or solution leading to the desired goal.

The connection between the procedure of decision-making and function-oriented description of a problem can help to reflect on the description of an uncertain situation. Reflection may be performed not only for the current situation, but also on any previously handled situation because the records explaining the design process are available and accompany the problem description. Current problem is also understood and described using knowledge models, therefore designer are not required to perform any extra work to record their decisions made during the design process. An justifications can be done mostly in a very simple form of link to the relevant portion of available knowledge base.

3.7 Summary

In Sections 3.2 to 3.6 I introduced several important strategies that may serve as a base for the knowledge-based support for the problem formalisation in design. The core of this approach is in using knowledge models for the description of problems that designers tackled in the past or are currently tackling. The implemented knowledge models are useful when designers attempt to describe the current situation using familiar terms from the previous situations. Knowledge models can provide them with 'a vocabulary' of known terms, as well as known techniques how similar problems were tackled in the past. During the initial stages of the problem formalisation designers recognise similarities between the problems on a higher level of abstraction. It was suggested in Section 3.5 that such an abstract level could be realised through the description of designed artefact's functions. Since it is possible to describe the functionality of an artefact without being committed to a particular structure, such description specifies the problem on a higher level of abstraction.

This proposal corresponds with Schön's observation that designers are able to discover similarities without being able to state specific attributes or components that. Schön (1983) describes in his book a situation when researchers were exploring why natural brushes perform better than artificial. After several trials one of them came with an idea that *natural brush behaves like a pump; when you press it on a surface, paint is 'pumped up'*. To

transform this observation into terms introduced in the previous sections, one can say that the researcher identified desired functionality of the problem, named it as *pumping*, and used familiar terms from this area to describe the behaviour of a brush.

Designers can apply knowledge models of already described problems to perform the desired shift of perspective. As it was described in Section 3.2 knowledge models are built on a top of common ontologies that provide a shared vocabulary for the description of various problems in a particular domain. Each problem in a particular domain is described using a hierarchy of such 'vocabularies' that conceptualise a problem on different qualitative levels. The functional descriptions of a problem may use mostly upper level ontologies. Behavioural and structural descriptions the lower levels of a problem description.

Once a higher-level problem description was finished (at least in a particular iterative step), designers can extend their current knowledge about a problem in question. They may explore the behaviour and the components describing the problem drawing upon knowledge from the retrieved problems that attained the same goal. When features identified within the current problem can be matched with the features of previous one, mapping rules can be calculated, and the supporting tool can provide other consequences of discovered analogy between two problems. Analogies can cause a shift of designer's attention to a particular sub-problem. Also they can cause that the current perspective and analogy are discarded, and trigger an attempt to discover new, more relevant analogies.

In my approach to the problem formalisation this activity can be supported through domain-oriented knowledge models that are built around various goals relevant for a particular domain. Descriptions of different problems can be linked to the different nodes in this 'goal-oriented functional ontology'. The link is interpreted as a description of a problem that attains the goal or functionality represented by a particular node. Similarly designers might define ontologies of behaviours and processes that lead to the achievement of a particular function. Different descriptions and their components will also be linked to the concepts from 'behavioural ontology'. The interpretation is that a component or a set of components has the particular behaviour that can further lead to the achievement of certain functions.

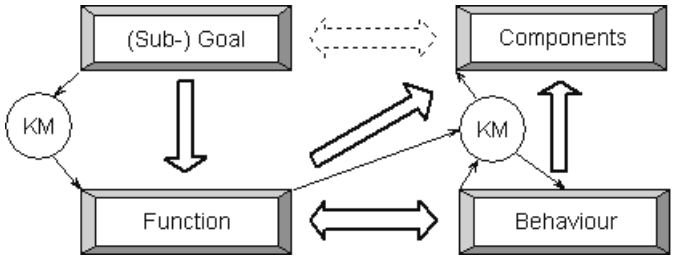

Figure 5. Typical procedure of problem formalisation

The entire procedure of an iterative problem formalisation using the goal-oriented approach is shown in Figure 5. 'KM' stands for knowledge model support; full arrows show the sequence of problem description steps, and dashed arrows represent reflection and possible changes in a higher level description. In such a way designers are able to describe the problem on three qualitatively different levels:

a) level of desired/achieved functionality (of an entire solution or its component);
b) level of desired/actual behaviour (of an entire solution or its component);
c) level of building components (problem decomposition)

The oscillation between the problem and sub-problems, and sub-solutions and the solution as depicted in Figure 6 (Cross 1997) can occur on any of the above-mentioned qualitatively different levels. It means that there exist alternative descriptions for a single problem, e.g. the functional description depends on a chosen goal, behavioural description is determined by the identified functions, etc. However, it is desirable to perform the 'decomposition' and oscillation rather on levels a) or b) than c). The scheme of such a general oscillation according to Cross for any of the above description levels is depicted in Figure 6.

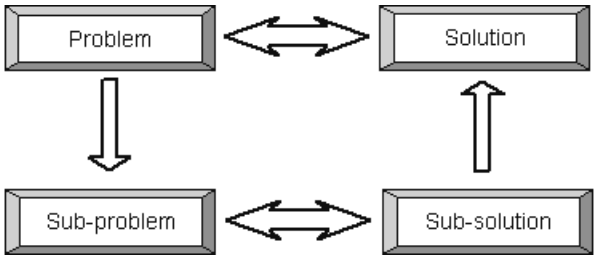

Figure 6. Oscillations during problem solving

To summarise, designers start their problem formalisation with a clear goal in mind. A known goal enables them to identify a set of known functions a problem may achieve within an uncertain design situation. Using knowledge-level models they can find out how a particular function (goal) can be achieved, and how a problem at hand may be described. Such approach reflects the fact that designers often draw upon their previous experience. Once they describe the functionality (goals) of the problem in familiar terms (components) they may encounter other issues that may change their current perspective, and help to identify further sub-goals, or modify the existing goals within the problematic situation at hand. Thus designers work in cycles – they reflect on their current problem understanding and description; the reflection may possibly lead to the problem re-structuring, or new description.

4 Summary and discussion

In the previous sections I reviewed some of the current research activities focusing on the processes occurring during early phases of engineering design. I referred to these phases as a problem formalisation, distinguished two major activities informing this phase – problem recognition and description, and showed that they are very closely linked with the solutions. To summarise, the most important feature of problem formalisation process is that it occurs in the conditions of uncertainty, ambiguity, and uniqueness of design situations. Due this feature it is difficult to describe the process of problem formalisation with a simple algorithm. Contrary, designers must often construct new 'problem-oriented' theories to describe and understand the current situations based on their deep theoretical knowledge and experience from formalising similar problems.

4.1 Scope of further research

Because the initial problem description is usually vague, informal, incomplete, and contradictory the formalisation and refinement (completion) of the initial description are important steps in the design process. As stated earlier, they must be considered at least as important as the generation of a solution satisfying known requirements. The following milestones of a typical design process and iterative nature of actions typically performed by designers are described in Figure 7:

a) **I –** an initial problem specification;
b) **F –** semiformal specification used at the input of PSM (e.g. a software package);
c) **A –** full, formally complete specification of an artefact.

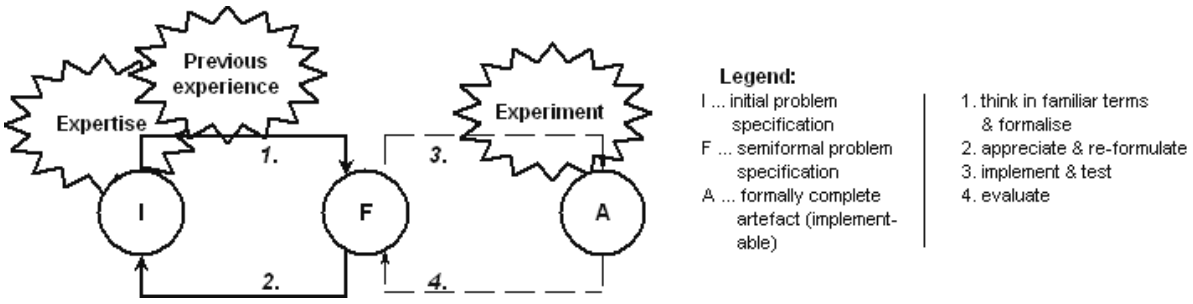

Figure 7. Typical milestones and actions in engineering design

The *initial problem specification* represents in real-world practice designers' initial ideas about the designed artefact or requirements on its functionality. The term '*semiformal specification*' can be understood as a description of design problem expressed in particular domain specific terms, containing variables, parameters, and sets of allowed values. It contains details about the designed artefact's functions and suggests how the desired functionality might be achieved, what are the alternatives, etc. *Formally complete specification* stands for a solution that satisfies given set of requirements from previous steps. Left-hand side of the scheme (full arrows) represents the problem formalisation cycle; right-hand side (dashed arrows) represents the problem solution cycle. Here belong implementation and evaluation that are influencing the problem formalisation cycle; however, these actions are beyond the scope of my current research.

In the proposed research I will address and explore the cycle between steps 1 and 2 as depicted in Figure 7. These steps correspond to designers' actions that were discussed in the

previous sections. Action labelled as '1' defines the perspective and concepts within an uncertain situation; in other words it is a problem formalisation consisting of problem recognition and description in domain-specific terms. When the current description does not reflect the situation correctly designers appreciate its impact on the current perspective, and may change it. This activity was introduced as reflection, and is shown as an arrow with a label '2'.

4.2 Discussion on the nature of design

Similarity- and analogy-based thinking has been identified as an important approach how to tackle uncertain situations in many different areas of human interests, including design. Designers typically attempt to describe the problem in familiar terms and when an unexpected result occurs, they begin to reflect on their current approach. Reflection was introduced as a process where designers are reviewing performed steps in their problem description and problem solution. Using reflection designers attempt to make explicit the decisions that led to an unexpected result and try to view the design situation from a different perspective. Therefore the process of formalisation has been described as inevitably intertwining and iterative process of problem recognition, description, solution, and reflection.

The iterative character of the problem formalisation and solution was clearly confirmed by other researchers who have studied activities in engineering design. Cross (1997), and Candy and Edmonds (1996) observed designers involved in creative design, and agree that an important feature informing design is designers' insight by which they understand the current uncertain situation. In both articles there are suggestions for a computational support of creative designers. According to Fischer (1992) the problem formalisation phase is presently far less supported and explored than the problem implementation. Conclusions from Cross, Candy and Edmonds, Fischer, and Schön are that the problem formalisation is nevertheless area worth of further investigation.

This observation is also confirmed by Altshuller (1984). He emphasises the phase of problem description and understanding and claims that once a problem (in his case a contradiction causing unsatisfactory performance of an artefact) is identified, it may be possible to solve such problem by an application of one or more 'inventive principles'. The principles represent generic descriptions and solutions, and must be further transformed to a domain-specific language before implementation and assessment.

All reviewed researchers agree that a creative insight or a contradiction discovery depends quite heavily on knowledge that is available during the design process. It may be intuitive (tacit) knowledge, experience, 'enlightenment', or a set of heuristic rules that narrows a vast space of possible solutions. This knowledge is made explicit by designer's reflection on the situation and iterative construction of problem description and solution.

4.3 Discussion on the importance of knowledge models

In the second part of this paper I reviewed the features of knowledge modelling and application of knowledge models built on top of re-usable ontologies in engineering design. I showed that knowledge models and ontologies could be beneficial in the knowledge-based support for the problem formalisation in design. The following features of ontological conceptualisation of uncertain situations have been found as the most important based on the research of Chanrasekaran *et al.*, Gruber *et al.*, van Heijst *et al.,* and Iwasaki *et al.*:

1. Clarification of the structure of knowledge about a particular problem, as well as particular problem domain;
2. Hierarchical description of the problem with different levels of abstraction;
3. Re-use of knowledge from previous designs and knowledge sharing within (possibly distributed) design teams.

A problem can be described using particular domain ontologies on multiple levels. Chandrasekaran *et al.* (Chandrasekaran 1993; Iwasaki, Fikes *et al.* 1993), and later Qian and Gero (1996) have identified three important levels of a problem description: structural components, their behaviours and relations, and functions as an effect caused by the relationships. They suggest that in practice designers should develop the functional description first, then try to locate different structures that can either directly or through behavioural interactions attain the desired functionality. Gero *et al.* point out that the functional description of an artefact (problem) can be done without any commitment to a structure, and might be useful when discovering analogies between two problems.

Functional description of a problem corresponds with the requirements that are set on the designed artefact, whereas structure can be understood as a solution satisfying the given set of requirements. Both descriptions are made using different descriptive languages and concepts. The concepts are organised in a form of common ontologies that enable designers to describe the desired functionality of their current problem (~ requirements) consistently and unambiguously. The same ontologies can serve as references when looking for an analogous problem, i.e. a problem where the same or similar requirements were recognised. Since there is a strong causal relationship between functional requirements, behaviours, and structures, it is possible to re-use the functional, behavioural, or structural description of a previous problem, its requirements or solutions, as a template for the current one. Of course such analogy might and might not work. It is on the designers to make the final decision whether the discovered similarity suits their current situations. Ontologies and knowledge models may significantly simplify the decision procedure, clarify the structure and relationships between chunks of knowledge, provide some reasoning from domain knowledge and justify it but the final decision remains on a human designer.

Ontologies are also useful for the support of reflection on the problem formalisation. They support the problem description in a standard, clearly structured form using ‘a common vocabulary’. Therefore reflection does not have to occur immediately after the problem formalisation and solution generation; it can be triggered later when a new unexpected result emerges. Ontology-based description of a problem enables other designers to review the work of their team colleagues without loosing important information about the problem, the design situation context, and the processes that led to the current description and solution. All activities designers performed during problem formalisation are documented using concepts from the problem-specific knowledge models developed on a top of common ontologies. Decisions are justified either by an existing domain model (~ domain theory), previously solved problem (~ experiential knowledge), heuristic rule from a base of best practices, or by a newly developed model for the particular situation that was constructed in accordance with the existing knowledge.

4.4 Research questions

This paper attempted to review the existing directions in the research of knowledge-based design support systems. It emphasised the co-evolving nature of design requirements and

solutions and the importance of support to problem description and reflection. In my further research I would like to address the following research questions and hypotheses[7]:

- **Question 1**:
  How can designers benefit from using knowledge-based support for the problem formalisation (PF)? How can be such support realised?
  Rationale:
  As it was mentioned there are many tools supporting designers in the implementation of already described problem. However, there is little systematic study and support for the problem description itself.

- **Question 2**:
  What are the roles ontologies may play in a specific application area of engineering design and PF?
  Rationale:
  Traditional role of ontologies is to clarify the structure of knowledge. It is a crucial feature for the clear description of a problem. Systematic structure of problem-specific knowledge may simplify reflection on current description of a problem. What are further implications of clear structure of problem knowledge?

- **Question 3**:
  How can be ontologies applied for the discovery of analogous problems that is a preferred professional strategy to tackle problems in engineering design?
  Rationale:
  There is a lot of research on how to map similar features between two problems that were found analogous; mostly they are based on the structural similarity. Actually, problems mapping can occur after the previous; probably most important phase was done. Problems mapping must be preceded by the discovery of potential analogy.

- **Question 4**:
  How is it possible to record co-evolving requirements and solutions together with their justifications?
  Rationale:
  It is quite straightforward to record and present final solutions to a problem, however these solutions will represent only a trifle of knowledge used to construct a particular solution. It is important to capture not only starting and ending point of the design process but also the evolution and history of decisions, together with their justifications.

[7] More details on the proposed technology of knowledge-based support using knowledge models are presented in a separate document that is based on this review.

5 References

Altshuller, G.S. (1984). Creativity as an Exact Science. USA, Gordon & Breach Science Publishers.

Bhatta, S. and Goel, A. (1994). "Discovery of Physical Principles from Design Experiences." Artificial Intelligence for Engineering, Design, Analysis and Manufacturing **8**(2).

Bhatta, S., Goel, A. and Prabhakar, S. (1994). Innovation in analogical design: A model-based approach. 3rd Intl. Conference on AI in Design (AID'94), Lausanne, Switzerland.

Brown, D.C. and Birmingham, W.P. (1997a). "Understanding the Nature of Design." IEEE Intelligent Systems & their applications **12**: 14-16.

Brown, D.C. and Birmingham, W.P. (Eds.) (1997b). IEEE Journal on Intelligent Systems & Their Applications (special issue on AI in Design).

Bylander, T. and Chandrasekaran, B. (1985). Understanding Behavior Using Consolidation. 9th Intl. Joint Conference on AI (IJCAI'85), Los Angeles, California.

Candy, L. and Edmonds, E. (1996). "Creative design of the Lotus bicycle: implications for knowledge support systems research." Design Studies **17**: 71-90.

Chandrasekaran, B. (1993). "Functional Representation: A Brief Historical Perspective." Applied Artificial Intelligence.

Chandrasekaran, B., Goel, A. and Iwasaki, Y. (1993). "Functional Representation as Design Rationale." IEEE Computer **26**(January (1)): 48-56.

Chandrasekaran, B., Josephson, J.R. and Benjamins, V.R. (1998). The Ontology of Tasks and Methods. 11th Banff Knowledge Acquisition for KBS Workshop, Canada.

Chandrasekaran, B., Josephson, J.R. and Benjamins, V.R. (1999). "What Are Ontologies, and Why Do We Need Them." IEEE Intelligent Systems & their applications **14**(1): 20-26.

Cross, N. (1997). "Descriptive models of creative design: application to an example." Design Studies **18**: 427-440.

Dominowski, R.L. and Dallob, P. (1995). Insight and Problem Solving. The Nature of Insight. R.J. Sternberg and J.E. Davidson. USA, MIT Press**:** 33-62.

Falkenhainer, B., Forbus, K. and Gentner, D. (1989). "The Structure Mapping Engine: Algorithm and Examples." Artificial Intelligence(41): 1-63.

Fischer, G. (1992). Domain-Oriented Design Environments. 7th Knowledge-Based Software Engineering Conference (KBSE'92), IEEE Computer Society.

Garrett, J.H., Jr. (1998). "The computer-aided engineer: Prospects and risks." Artificial Intelligence for Engineering, Design, Analysis and Manufacturing **12**: 61-63.

Gentner, D. and Forbus, K. (1991). "MAC/FAC: A Model of Similarity-based Retrieval." Proceedings of the Cognitive Science Society.

Gero, J.S. (1990). "Design prototypes: A knowledge representation schema for design." AI Magazine **11**(4): 26-36.

Gruber, T.R. (1993). "A Translation approach to Portable Ontology Specifications." Knowledge Acquisition **5**(2): 199-221.

Iwasaki, Y., Fikes, R., Vescovi, M. and Chandrasekaran, B. (1993). How Things Are Intended to Work: Capturing Functional Knowledge in Device Design. International Joint Conference on Artificial Intelligence.

MacCallum, K.J. (1990). "Does Intelligent CAD exist?" Artificial Intelligence in Engineering **5**(2): 55-64.

Motta, E. (1997). Reusable Components for Knowledge Modelling. Knowledge Media Institute. UK, The Open University.

Motta, E. and Zdrahal, Z. (1998). A principled approach to the construction of a task-specific library of problem solving components. 11th Banff Knowledge Acquisition for KBS Workshop, Canada.

Mulholland, P. (1999). The Enrich evaluation strategy, KMi, The Open University.

Nakakoji, K., Sumner, T. and Harstad, B. (1994). Perspective-based critiquing: helping designers cope with conflicts among design intentions. AI in Design'94.

Newell, A. (1982). "The knowledge level." Artificial Intelligence **18**(1): 87-127.

Nidamarthi, S., Chakrabarti, A. and Bligh, T.P. (1997). The significance of co-evolving requirements and solutions in the design process. 11th ICED, Finland.

Prabhakar, S. and Goel, A. (1998). "Functional modeling for enabling adaptive design for new environments." Artificial Intelligence in Engineering **12**: 417-444.

Qian, L. and Gero, J.S. (1996). "Function-Behaviour-Structure Paths and Their Role in Analogy-Based Design." Artificial Intelligence for Engineering, Design, Analysis and Manufacturing **10**: 289-312.

Schön, D.A. (1983). Reflective Practitioner - How professionals think in action. USA, Basic Books, Inc.

Simon, H.A. (1973). "The structure of ill-structured problems." Artificial Intelligence **4**: 181-201.

Smithers, T., Conkie, A., Doheny, J.*, et al.* (1990). "Design as intelligent behaviour: an AI in design research programme." Artificial Intelligence in Engineering **5**(2): 78-109.

Sveiby, K.E. (1997). Tacit Knowledge. WWW page, current at 7/4/1999, at http://www.sveiby.com.au/Polanyi.html.

Swartout, W. and Balzer, R. (1982). "On the Inevitable Intertwining of Specification and Implementation." Communications of the ACM **25**(7): 438-440.

Tang, M. (1997). "A knowledge-based architecture for intelligent design support." Knowledge Engineering Review **12**(4): 387-406.

Thompson, G. and Lordan, M. (1999). "A review of creativity principles applied to engineering design." Journal of Process Mechanical Engineering **213**(1): 17-31(15).

van Heijst, G., Schreiber, A.T. and Wielinga, B.J. (1997). "Using explicit ontologies in KBS development." International Journal of Human-Computer Studies **46**(2/3): 183-292.

Watson, I. and Perera, S. (1997). "Case-based design: A review and analysis of building design applications." Artificial Intelligence for Engineering, Design, Analysis and Manufacturing **11**: 59-87.

Wielinga, B. and Schreiber, G. (1997). "Configuration-Design Problem Solving." IEEE Journal on Intelligent Systems & Their Applications **12**(2): 49-56.

Appendix A

Scenario of knowledge-intensive formalisation

In this section the functionality of the proposed knowledge-intensive approach for the problem formalisation will be illustrated. It is done in form of a scenario that reflects procedures a designer performs in the domain of dynamic systems modelling and simulation.

The designer's task is to develop a model that suitably reflects the behaviour of a device at hand. The starting and the ending points of the design process are depicted as (a), and (b) sections in Figure 8. Let us assume that designer made a sketch of the layout and connections between the system components as depicted in Figure 8(a). The sketch can be understood as a simplified, initial description of a real dynamic system. In a real-world practice it may correspond to the initial customer's requirements, and/or ideas of the designed artefact. An engineer can prepare such a sketch at the customer's site visit, for instance.

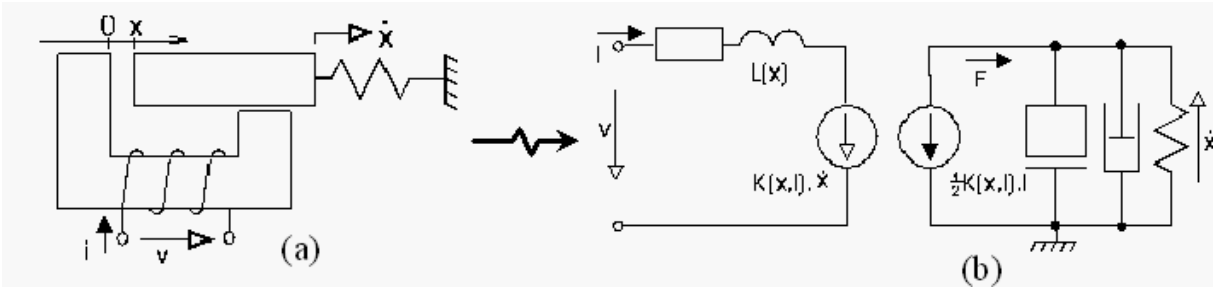

Figure 8. Basic steps in dynamic systems modelling:
(a) system identification, (b) iterative development of a model

Let us assume the designer observed (or wants to achieve) the following behaviour: when electric current flows through the circuit the distance marked as '**x**' decreases. The larger current flowing in the electric circuit the faster '**x**' decreases, the anchor is attached to a spring that causes counter-effect in accordance to decreasing '**x**'. To model such a device designer starts to think of it in familiar terms. For example the starting point could be the following thought: "*The behaviour of this device reminds me a pendulum, … the more I elevated a ball of a pendulum, the more power I had to use to achieve it, … I mean this device here acts also as some oscillating element …*". Designer's attention is focused on the device behaviour that reminds a familiar problem from the past. With words "*as some oscillating element*" s/he attempts to locate the perception of similarity to the previous problem, and simultaneously recognises the problem but its description is still vague. Designer might be supported in this phase in form of a domain-specific model that contains various informations about problems that have similar goals that were recognised similarly.

Now we can imagine designer recalls the situations that concerned *some oscillating elements*. In this phase we can support him/her with retrieving the situations that contain the same concept (an *oscillating element* in our case). It is possible to use a knowledge-level model of problems within a particular domain as a reference for the retrieval of 'similar and familiar' situations. We assume that the previously solved problems were described using knowledge-level models of a particular domain. Our designer found a pendulum problem that was modelled in the past. According to the actions depicted in Figure 7 the recognition of a problem is followed by the appreciation of similarity of the two situations. Appreciation

might be performed as follows: "*... just in the case of a pendulum I had to overcome gravity with my force, ... there is always a damping force on one side and a counter-force on the other side. Let me model this movable element of the device as a pendulum, and a spring as a damper, ... it seems right, ... spring as a source of counter-force ... it might correspond to gravity force…I have to move from mechanical systems to electric circuits"*

The previously solved problem serves as a familiar framework designer attempts to set on the current problem. As we see, there was an immediate attempt to commit to the same strategy as in the previous problem. Description of this strategy was included within the retrieved case. But there was 'surprising' fact that it did not solve the entire situation, because so far nothing was found and described to counter-effect the damping force of a spring.

Designer's experiment continues: "*Right, let me see the rest of this device, this electric current ... it must be the source that acts against the counter-force. Looks as if the change in current here generated an effect in pulling the anchor over here ... It means electric energy must be somehow transformed on a mechanical…*". As stated earlier, when a designer is surprised by the result of the performed action a different perspective to view the problem may be applied; a different part of a problem can be focused on. This strategy is clearly visible in the previous sentences. Our designer focused on finding a source that acts against spring, and realised that there must exist a transformer to transform the electrical energy on the mechanical one. To follow this strategy, and see the effects of this decision designer has to enhance the initial knowledge about the problem as well as existing problem description. Two kinds of energy interactions and a transformer concept are introduced.

As the next action, the designer may, for example try to explain the process of transformation, what influences the transformation, and what is the relation between electrical and mechanical quantities. The process of transformation must be described to see if the model performs similarly as the real device or a device expected by a customer. In this stage a knowledge-intensive support can be useful to avoid 're-inventing the wheel'. Simply, a designer can be presented with several cases that implemented a concept of transformer, possibly electrical vs. mechanical transformer. It will help to find out the appropriate quantities influencing the transformation process, and/or identify the relationship among them more straightforward.

A supportive system can also provide designers with the alternative solutions and approaches. It can be based on their similarity of the effect, or behaviour with the current solution. Using a knowledge-level model designer can efficiently look for alternations to the current problem description. Alternative descriptions might be explored in parallel, as described in Section 2.2. Knowledge-level model serves as reference for the description of significant features of various models of a particular device.

Earlier I claimed that a support system must be interactive. It means that the communication between the designer and a support system must be bi-directional. So far mostly the information flow from a support system towards the designer was discussed. Let me discuss briefly the opportunities for the designer to interact. The interactions will be of two major types basically – current problem description, and questions about previous problems. In the former type of interaction the designer defines the perspectives to view the problem, and identifies concepts describing the problem. Very useful feature for the future re-use would be recording of the designer's decisions and justifications of the performed steps. Problem description in conjunction with justification could serve as rationales behind the undertaken strategy.

In the latter type of interaction the designer may take the control over the entire process of support. S/he may ask the supportive system for similar solutions, or alternatives that satisfy specific criteria, instead of waiting for the system to help. Thus two different modes of operation will be desirable – automatic support control (computer has initiative) vs. support on designer's demand (human designer has initiative). Using such an approach the support system can be easily customised for the different levels of expertise. Novices may use more automatic support; in contrast the experts may take initiative themselves.