Supporting Engineering Design Using Knowledge Based Systems Technology with a Case Study in Electricity Distribution Network Design

A thesis submitted for the degree of Doctor of Philosophy

by

Janet Theresa McDonnell

Department of Computer Science, Brunel University

February 1994

In memory of my parents,

Dennis McDonnell

and

Bernadette McDonnell

ACKNOWLEDGEMENTS

Firstly, I thank Professor Leslie Johnson who has nurtured my education, for providing a stimulating learning environment, and for helping me to acquire the skills to benefit from it.

Secondly, I thank John Holland of London Electricity PLC for giving me so much of his time to talk about his work. I also thank Bill Cartwright, also of London Electricity, for introducing me to John Holland and for negotiating on my behalf in the Engineering Director's Department to which I am grateful for permission to carry out the case study.

I also thank Brunel University for the award of a grant to support some of this work and to the Department of Computer Science in particular for the resources to produce this thesis.

I also gratefully acknowledge the following for their help:

Brian Logan and Tim Smithers who generously supplied me with copies of published and unpublished material on the Edinburgh Designer System described in chapter 3;
Andy O'Keeffe, for comments on drafts of the chapters in part 1 of this thesis;
Elpida Keravnou and John Washbrook who gave me initial encouragement to continue my formal studies beyond MSc level;
my colleagues at Brunel, Galal Hassan Galal, Nancy Johnson, Sarmed Al Shawi and Tony Elliman for various, vital bits of information or advice and also for their general encouragement throughout this work.

Finally, a personal thank you to my family, to my daughter Heather Sinclair, and to my brothers Ian and Robert McDonnell for support and encouragement in this work and in everyday life, and last, but far from least, thanks to Andy O'Keeffe for enduring my interminable accounts of every dot and comma and all the dramas associated with each one of them.

Janet McDonnell,
February, 1994.

ABSTRACT

This thesis explores the architectural requirements of engineering design support systems based on knowledge based systems technology. The exploration is based on an understanding of the nature of designing as a professional activity and on the extent to which designers' competence can be modelled. Attention is focused on certain salient aspects of designers' competent behaviour. The theoretical study leads to the specification of requirements to be satisfied by a knowledge based system which will support designers in their professional setting and to the proposal of some knowledge based system components which will meet the requirements identified.

The theoretical aspect of the thesis is complemented by a case study based on a designer of high voltage electricity distribution networks. The case study illustrates the theoretical component of the thesis and the methodological basis for the work. The practical realizability of the components of the knowledge based systems architecture proposed are demonstrated using the results of the analysis of the knowledge elicited in the case study without prejudicing the general applicability of the ideas. An object-oriented knowledge engineering software development environment is used to demonstrate how some components of the design situation represented can be implemented.

CONTENTS

FIGURES

APPENDICES

CHAPTER 1

Introduction

"Let us carefully devise and direct our acts, for the past we create for ourselves shapes our destiny, determines it, and gives us our inheritance. I would say that Fate exists, but that we are responsible for it."

Amédée Ozenfant

This thesis explores the architectural requirements of design support systems based on knowledge based systems technology. The exploration is based on an understanding of the nature of designing as a professional activity and on the extent to which designers' competence can be modelled. Attention is focused on certain salient aspects of designers' competent behaviour. The theoretical study leads to the specification of requirements to be satisfied by a knowledge based system which will support designers in their professional setting and to the proposal of some knowledge based system components which will meet the requirements identified. The theoretical aspect of the thesis is complemented by a case study based on a designer of high voltage electricity distribution networks. This empirical study is an integral part of the thesis, since all of the work presented here is grounded in the belief that any useful design support system must be based on a thorough understanding of the designer's task, as he sees it, and on an appreciation of the professional setting within which designing takes place. The case study permits the theoretical component of the thesis and the methodological basis for the work to be illustrated and allows the practical realizability of the components of the knowledge based systems architecture proposed to be demonstrated without prejudicing the general applicability of the ideas.

1 Research Context

1.1 Background

Research into developing second generation knowledge based systems founded on models of competence (Keravnou, 1986; Johnson, 1986; Johnson, 1986a) has occupied researchers at Brunel University for a number of years. Work to develop a knowledge based systems architecture which was first applied to fault diagnosis (Keravnou, op.cit.) has been followed by related studies extending the original ideas in the direction of intelligent data handling and distributed knowledge bases among others (Murdoch, 1990; Stylianou, 1991). The broader methodological aspects of competence modelling also have been pursued particularly in the area of knowledge elicitation (for example Graham,1990; Tomlinson, 1993; Funes, 1994). This thesis is concerned with applying competence modelling to engineering

design and with interpreting and extending the work on the Competent (Expert) Systems architecture for the support of engineering design.

Conventional computer aided design (CAD) tool development has been driven by the aim to reduce the amount of effort which has to be put into design. The degree to which a CAD tool aids design is judged therefore by the extent to which it automates some routine, tedious, or otherwise complex but nevertheless unambiguously specifiable, aspect of design work. CAD tools have tended to be specific and narrowly focused to satisfy the requirement to automate or dramatically reduce the amount of human effort required to achieve some very specific task.

Some of the research aimed at extending CAD tools to make them more flexible or more general is referred to as intelligent CAD (ICAD). It is characterised by a movement "outwards" from specific tools which already automate some particular design activity towards more generality, interconnectivity, or flexibility. The advantage of extending what CAD tools can do by this route is that there is always some core functionality supplied by the CAD tool. The disadvantage of this approach is that strategic issues receive low priority and the increasing complexity of the tool's operating environment (as the boundary of its scope moves outwards) is catered for piecemeal in an ad hoc fashion with the attendant inevitable radical rethinking being needed whenever a major step forward is to be made.

One prominent alternative approach starts with A.I. techniques and seeks to apply them to the support or automation of design activities. Researchers, mainly from the "A.I. community", working from this perspective, look towards applying the practical aspects of A.I. research to the "problem" of design. Green (Green, 1993) has associated the acronym KAD (knowledge aided design) with the body of work of this kind which applies knowledge based systems research in particular, to applications associated with designing. Some of the work based on this approach is concerned with producing CAD tools using A.I. techniques. Thus the range of computer aided design tools is extended to applications where the "traditional" techniques associated with CAD (e.g. those used for geometrical modelling or image storage and retrieval) are too limited or are inadequate in some way to provide the support needed. The work presented here, although it can accurately be described as being concerned with knowledge aided design (and therefore is, in some sense at least, KAD), does not share the approach or central concerns of this branch of KAD. Although it is true that knowledge based systems technology and not a specific CAD tool forms the technological starting point for providing the design support explored here the core motivation for the work, and the basis of what is proposed lies elsewhere than in applying A.I. techniques to produce enhanced CAD systems. Rather, it is believed that understanding and modelling professional expertise is the key to how knowledge based systems technology should be exploited to support designing, and that applied on this basis it can make a key contribution, at a strategic level, to the support of practising expert designers.

The approach taken demonstrates that knowledge based systems technology has the potential to provide a framework for supporting design. Within this framework specialised CAD tools, conventional databases, intelligent data handling, traditional computer programs for analysis, simulation, etc., can be integrated, each continuing to make its particular contribution to automating or assisting aspects of designer's work set in a context which is relevant, timely, and meaningful to the designer. This thesis argues that a knowledge based system able to provide a strategic framework such as this which can co-ordinate and make use of heterogeneous, distributed design resources must be structured on the basis of a model which accurately reflects the designer's task.

1.2 Perspective

The work presented here is grounded in the belief that an understanding of what the designer is trying to do, what he considers important about a design situation, and what he perceives his design task to be, is a fundamental prerequisite for producing any sort of knowledge based design support system to assist a designer's work. Whilst at first sight this statement may appear to be a statement of the obvious, even a superficial survey of the literature describing application of A.I. techniques to design applications will reveal that many of the systems reported fall into the category of what Dreyfus (Dreyfus, 1981) has termed micro-worlds[1] applications rather than sub-worlds applications which have some intrinsic legitimacy and possible potential for scaling up to wider or more complex situations.

Unfortunately, it is the attraction of the micro-worlds approach - its permitting of a domain to be defined that can be analyzed in isolation - that is the heart of its weakness for, by the very means it employs to handle the complexity of situations, it eliminates that which gives meaning or makes sense of the objects or activities it claims to define and represent[2].

An example from the field of architectural design, the design of domestic architecture, illustrates the radical differences between, on the one hand, setting out to investigate design support using a micro-world approach, and, on the other hand, basing one's efforts on empirical study of designers' practices. The figure shown as figure 1.1 is given as a design space decomposition for the task of designing a house. The figure is taken from the introductory chapter of a recently published three volume work on artificial intelligence in engineering design (Tong, 1992). It is presented as

---

[1]The first use of the term "micro-world" is attributed to Minsky and Papert, but here it is used with the negative connotations ascribed to microwords by Dreyfus. A micro-world is a fairyland in which things are so simplified that almost every statement about them is false in the real world.

[2]Dreyfus discusses this aspect of the stagnant nature of A.I. research based on micro-worlds at length in Dreyfus, 1981.

an example of a design space represented as an and/or tree. The authors make no explicit claims as to its validity, on the other hand they make no disclaimer. In any case, it is the example's status as an accurate exemplar of an approach to A.I. in design which is important here; the purpose of using this particular example is not to ascribe responsibility for it to particular individuals.

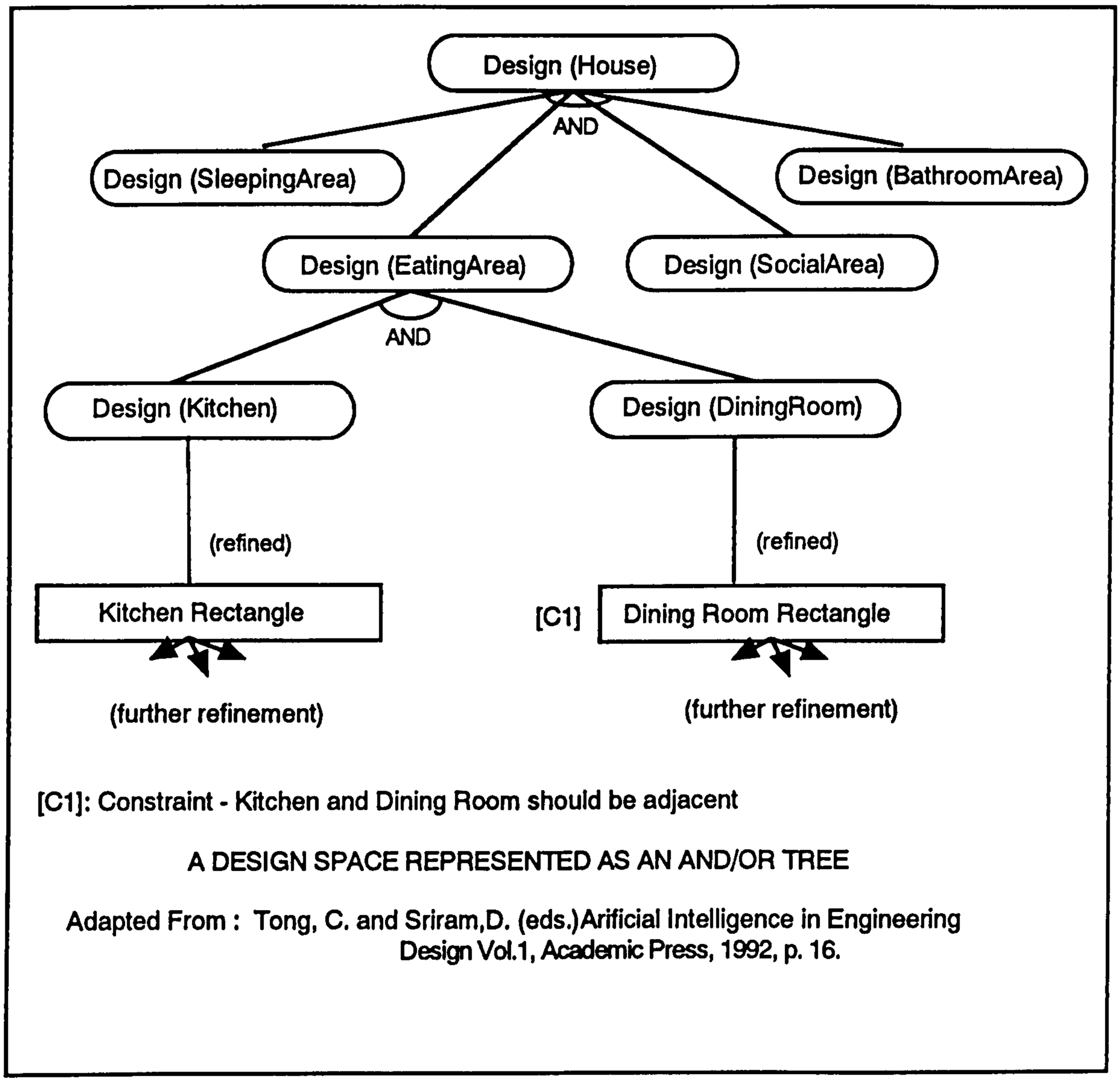


Figure 1.1 Typical example of an "A.I. approach" to representing design problems.

The figure presents a typical approach to design problem decomposition, that is a decomposition which is based on the component parts of *a design solution*. The figure portrays the design of a house as involving the design of four (living) areas, each of which may be designed in turn by further decompositions along the same lines. The further decompositions are arrived at (also) by further explicit or implicit consideration of the use

which is to be made of different areas of the house. The figure also shows that some constraints apply, these are an acknowledgement that there are interrelationships between parts of the decomposed design. These are shown as links between some of the nodes in the tree which represents the design space. Thus, nodes are linked, both by a decomposition hierarchy and "orthogonally" by constraints (such as the requirement to put the kitchen adjacent to the dining room shown in figure 1.1).

The first thing to note about figure 1.1 is that it does not represent a design *problem* decomposition at all, rather it shows a decomposition of a design *solution*. Any human designer or A.I. system designing a house in the way depicted by the representation shown will only produce a very limited variety of house designs. Researchers taking an approach to A.I. support for design which accords with this example counter this line of criticism by defining a type of design (often termed routine design) which, by definition, is the type of design this class of A.I. systems carries out.

The topic of routine design is dealt with later in this thesis, in particular, one A.I. system for routine design is described at length in chapter 3. Here, however, an argument about the existence, validity, nature, and value of routine design, however it may be defined, is not the most fundamental objection to the design space decomposition shown in figure 1.1. It is the objection to micro-worlds which is the most overwhelming. The clue to the problem with the representation of housing design shown in figure 1.1 cannot be seen by examining the decomposition it depicts precisely because the objection lies beyond the micro-world it represents.

The point can be made rather dramatically by calling upon some rather famous examples of housing design. Consider Frank Lloyd Wright's Falling Water or Mies Van Der Rohe's Farnsworth House, or even the public housing scheme at Byker by Ralph Erskine. The most striking aspect in common in each of these very different designs is the relationship of each building to its *surrounding* site . One of the major considerations in housing design - one of the major challenges for the designer - is in making effective use of the site. (The budget, something else which does not feature in the micro-world, is another major influence on what the designer does.) The examples chosen to make the point are, admittedly, rather extreme examples. However, empirical investigation into what those who routinely design (more mundane) housing actually do has revealed the following:

> "Unlike many other buildings the architect may design, the house has an internal structure which is relatively simple and easily understood. What makes the *internal planning* of an individual house difficult however is the problem of relating it to adjacent houses and other features of the site ... Perhaps it is this very close and *critical interplay between internal and external constraints* which makes housing such a fascinating but difficult design problem. It certainly seems likely that the *balance of internal and external constraints* in a design problem is of considerable significance in determining the *nature of that problem* and the

designer's response to it"

(Lawson, 1990, p.76, emphasis added)

It is clear from this that a study of what house designers' actually do *completely undermines* the status of the representation of house design shown in figure 1.1. That which has been left "outside" the boundary of the micro-world which figure 1.1 represents is exactly that which determines what any valid representation of the design problem space should contain.

This thesis starts from the premise that any system that is going to assist or support a designer must make sense to him, in the sense that it must be based on an understanding of what is important to the designer and relevant to the problems he (really) faces when designing. An understanding of the designer's perspective will have inevitably, a formative influence on the design of anything which is to assist him with his designing. What the designer sees his task to be must have a direct bearing on the form and structure of any useful design support system.

2 Aims

The aim of this work is to explore what is needed for successful partnership and co-operation between a designer and a knowledge based system which is capable of supporting him in his work. More specifically, the intention is to bring understanding of the nature of design as a human social activity , understanding of the powers and limitations of models of expertise, and understanding of the capabilities of second generation knowledge based systems technology to bear on the task of supporting engineering design. The purpose of the work is to explore the relationship between design competence and professional design practice with a view to understanding how a support system can fit into the professional environment which forms the context for the designer's work. The thesis aims to examine the ways in which focusing attention on the designer's perspective (what he sees his work to be, and how he views the professional setting) affects the way in which knowledge based design support systems should be developed, the qualities they need to exhibit, and the way they should be designed to achieve these qualities. An important further aspect of the work is to consider the inter-relationship of these issues.

For the reasons set out in section 1.2 above it is believed that it is essential not to shy away from studying real design activity in the natural setting of design practice. To achieve the aims of the work, to entertain the hope of doing something useful towards supporting designers using knowledge based systems technology, it will be essential to have an understanding of what the design task appears to be from a designer's perspective. A case study therefore forms an integral part of the thesis. It presents an opportunity to explore and test what are the best ways of eliciting what the design task entails and of developing knowledge based systems to support design based upon this understanding. The outcome of research with such an empirical element is inevitably unlikely to be as

"neat and tidy" as work based on something hypothetical or some problem which is otherwise constrained. However, this state of affairs is unavoidable if it is hoped that the result of the research will contribute to the understanding of where knowledge based systems technology can *realistically* make a contribution in the *real* world of the practising designer.

3 Objectives

A series of objectives have been identified to achieve the stated aims. Although they are separately stated, these objectives are strongly inter-related with one another. Whist the thesis can be seen to deal with each particular objective separately as the focus shifts among the aspects and issues being explored, it is also the intention to show how each focus affects and is affected by the others. Each objective defines an area of interest but at the same time the nature and composition of each area co-determines and sets in context each of the others. The separately distinguished objectives are to

- identify how the contributions of complementary studies of the nature of design as a human activity advance the overall understanding of it;

- review the spectrum of applications of that branch of A.I. research concerned with knowledge based approaches to design support, identifying problems and limitations which have been experienced in this field by focusing on some particularly distinguishing characteristics of design activity;

- analyze problems with knowledge based support of design in terms of the required components of a model of design competence on which to base a design support system;

- identify requirements to be met for supporting design practice and to relate these to facilities they necessitate in a knowledge based system for design support;

- undertake an empirical study of a designer's practice to develop and test interpretations of the design situation to explore how design in the domain studied can be effectively supported by a knowledge based system;

- identify and to apply an appropriate variety of knowledge elicitation, knowledge analysis, and knowledge representation techniques within a coherent methodological framework;

- demonstrate how elicited design knowledge may be interpreted in terms of a more general understanding of the nature of designing and of design practice (see earlier objectives);

- show by example how some of the essential components of a knowledge based system to support design (identified as a result of earlier objectives) may be mapped to a software implementation platform.

4 Organization of the Thesis

The remainder of this thesis is organized into two parts. Part 1, which consists of chapters 2 to 5, comprises the theoretical element of the work. Part 2 of the thesis, chapters 6 to 8, constitutes a case study in engineering design knowledge elicitation, analysis and modelling. Part 2 reports the investigation of the design expertise of an electrical engineer from the electricity supply industry responsible for the design of electricity distribution networks. Chapter 9 presents concluding remarks about the work reported in the thesis. Although there are different ways of reading the thesis (as described below in section 4.2) it is emphasised that each chapter in parts 1 and 2 is in one sense self-contained, in that each presents a sub-thesis of its own. However, the way in which the contribution of each chapter is presented is affected by its context which is given by all the other chapters and by its relationship to them. The thesis, read as a whole, thus presents a coherent synthesis of the contributing elements in which each chapter plays an important part.

4.1 Overview of Chapters

This chapter, chapter 1, has introduced above, the background to the research presented and the aims and objectives of the study undertaken and reported in the remainder of this thesis.

Chapters 2 and 3 give the background of theories about designing and approaches to supporting design using knowledge based systems technology respectively. Chapter 2 traces the development of ideas about the nature of design over the last three decades showing how the concentration of design theory research in different periods on different aspects of design has contributed to the overall understanding of design at the present time. Chapter 3 describes three approaches to knowledge based system support of design which are representative of the variety of research approaches and which between them cover the spectrum of concerns and issues in supporting design with knowledge based systems technology.

Chapter 4 discusses the modelling of designers' competence. It draws together theories on the nature of expertise, and the nature of models, and introduces the fundamental components of competent systems. Problems with knowledge based systems for design support are analyzed in terms of

the extent to which the models on which they are based reflect or relate to salient characteristics of designers' behaviour.

Chapter 5 explores the requirements for supporting design practice which are consequential upon the findings about how designers practice their profession. It relates the requirements which emerge to components of a knowledge based system to support design.

Chapter 6 introduces the empirical work undertaken. It describes the knowledge elicitation process in detail by describing the knowledge elicitation aids used and their roles in gathering, checking and analyzing the elicited data. It gives an account of the theoretical basis of each of the aids to knowledge elicitation used and explains the underlying methodological basis for the approach taken.

Chapter 7 presents an analysis and interpretation of the data elicited. The data is presented in two forms, firstly in terms of examples of constructed descriptions of the design process and secondly in the form of a representation of the aspects of the design situation which are relevant to competent performance on the part of the designer.

Chapter 8 demonstrates how some of the data elicited, analyzed and interpreted in the way shown by chapters 6 and 7 can be represented in components of a knowledge based design support system implemented on the sort of software development platform provided by an object oriented knowledge engineering environment.

Chapter 9 presents concluding remarks about the work described, making reference to both the theoretical study (presented in part 1) and the empirical work (presented in part 2).

4.2 Ways of Reading this Thesis

A number of approaches to reading the material presented are proposed as alternatives to the obvious complete, sequential path. The recommended reading paths through the thesis are illustrated in figure 1.2.

A reader who either has a thorough knowledge of the development of ideas about how designers design or who has little interest in this history may omit reading of chapter 2 altogether. However, there are assumptions that a reader of chapters 4 and 5 will have an understanding of this material and an appreciation of the way some of the material in chapter 7 is presented will be enhanced by a knowledge of what is presented in chapter 2.

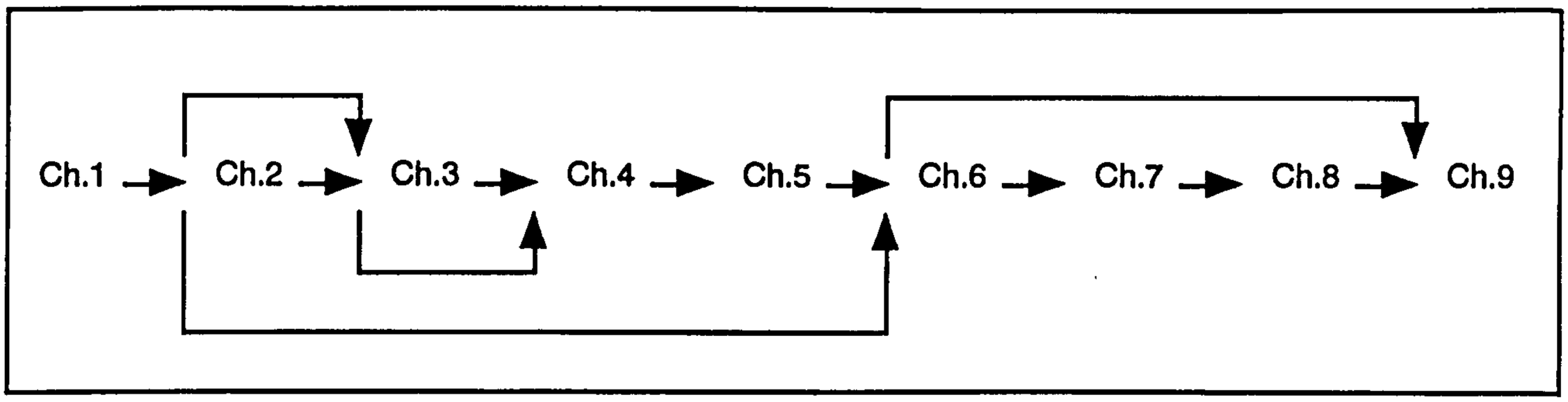


Figure 1.2 Reading paths through the thesis.

Chapter 3 may be omitted by readers who are either familiar with the systems it describes or who are already familiar with the spectrum of knowledge based engineering design support systems reported in the literature. There are some references to material in chapter 3 in chapters 4 and 5 where the systems described in chapter 3 may serve to illustrate some point raised or issue discussed. When such references are made the reader who has omitted reading chapter 3 in full may easily refer back to the relevant section of chapter 3 referenced or may skip over the illustration being given.

Readers who have no interest in the case study may read chapters 1 and 9 with the theoretical part of the thesis (chapters 2,3,4 and 5). Alternatively, the empirical investigation and practical work, the report of which forms the case study of the thesis (chapters 6, 7 and 8), can be read with chapters 1 and 9 in a complementary manner. However, inevitably, the concluding remarks which comprise chapter 9 make reference to both the theoretical and empirical parts of the thesis.

CHAPTER 2

Understanding Designing - A History of Design Theory

"He who appeals to authority when there is a difference of opinion works with his memory rather than with his reason."

Leonardo Da Vinci

This chapter traces the development over the last thirty years of ideas about what is called design, in particular that which is termed the design process. Thirty years of study has lead to a certain maturity in the field of design research; over this period the research has resulted in methods, models, and tools for designing which have been directly useful for specific design situations. If there appear to be more questions left unanswered now than there appeared to be thirty years ago, they are certainly less naive ones.

Cross, writing in 1984 about developments in design methodology from about 1960 onwards, analyzed the developments in terms of a movement through four stages which he characterized as; *prescription* of an ideal design process, *description* of the intrinsic nature of design problems, *observation* of the reality of design activity, and *reflection* on the fundamental concept of design (Cross, 1984). This observation, which reveals an interesting shift in the object of investigation, offers a useful framework within which to review the concerns and themes which have preoccupied design theorists at least as far as the mid-1980s. It is used here for that purpose. It will be seen that each stage influenced those that followed and each new emphasis has lead to a re-interpretation of the value and meaning of those that preceded it. Thus, each stage contributes something to the building up of a richer overall picture of what it is to engage in design. This is not to imply that we now have all the answers, or that design is now completely understood, or that designing can be expressed unambiguously in an algorithmic sense. On the contrary, current understanding indicates that not only are we far from this position but also that it is by no means obvious that we can ever attain it. The view may also be ventured that such a position is not even realistic or desirable.

1 The Design Methods Movement

The development of the design methods movement in the 1960s centred upon "concern for *systematic* procedures for overall management of the design process and on *systematic techniques*, called design methods, to be used within such a process" (Cross, op.cit.). The motivation for this movement was based on three related factors. Firstly, a desire to "make design thoughts public", that is to make designing and the designer's actions more open to inspection (Jones, 1991) and to discourage unjustified claims to artistic individuality (Alexander, 1964). Secondly, a need to respond to the increasing complexity, uncertainty, and instability associated with designing in and for a rapidly changing technological society. The response to this was to address both the separation of design from implementation (a contrast with craft-based approaches) and the division of a complex design task among cooperating specialists (Lawson, 1990). Thirdly, there was what amounted to fear of the apparently unscientific approaches that designers used to solve problems, as though this some how rendered them disreputable, a notion bound up with the idea that a narrowly scientific approach must be the basis of any respectable paradigm.

1.1 What Was Proposed

The emphasis, in this era, was on applying *rational* methods and *systematic* approaches to all design activities. Design methods were identified and presented uniformly as procedures to be applied in many of the separate stages of the design process. A key example of a publication in this area is Jones' text "Design Methods Seeds of Human Futures" first published in 1970 (Jones, 1970). Many of these so-called design methods were appropriated or adapted from disciplines “outside” design but seen as similar in some respects, so for example operational research and decision theory methods which had been applied with some success to planning problems provided some of the “new” design methods. Each method was associated with particular aspects of the design process. So, for example, a set of methods is associated with generating design ideas. Brainstorming (Osborn, 1957) and synectics (Gordon, 1961) are two examples in this category for systematising lateral thinking. For systematically comparing alternative design proposals on a quantifiable basis, the method of cost benefit analysis was advocated. For planning and controlling the design process one method proposed was critical path analysis. Collections of methods were assembled to cover aspects of designing, typically; generation of ideas, systematic searching, data gathering, morphological transformation (rearrangement of components within a design), evaluation of alternatives, and so on. Jones' text presented over thirty methods in a uniform "cook-book" style.

Alongside the enthusiasm for the use of *rational* methods, and intimately associated with it for the reasons already stated, there was a need to define design, in particular to define it as a scientific process, which being defined, could subsequently have its complexity managed. Individual design methods were to be applied, as we have seen, to particular design activities. This demanded that the design process be viewed as a set of stages or steps within which, at appropriate points, design methods are applied. Geoffrey Broadbent writing in 1979 (Broadbent, 1979) described the view of designing at this time as follows :

> "Most of the pioneering design methodologists discussed the nature of design as a science before proceeding to their personal descriptions of techniques which, hopefully, designers would be tempted to adopt in practice. And, almost without exception, they took a Cartesian view of designing; breaking the problem down to fragments and solving each of these separately before attempting some grand synthesis." (op.cit., p.337)

A desire to manage design activity and to be able to systematically relate design methods to parts of the design process demanded certain kinds of models of the design process. Whilst the details of these models varies from one source to another some general features can be seen. The design process is described as a sequence of stages (and/or steps). These are grouped into phases of design. Feedback, reworking, or iteration of some kind signifying opportunities for reconsideration and revision in the light of evaluation or new input of some sort is a feature of all of these models of the design process.

Asimow (Asimow, 1962), for example, offers a spiral model in which successive stages of design each involve steps characterised as analysis, synthesis and evaluation. Most procedural views of design include about four separately identified stages or phases. These can loosely be described as follows. The first is clarification of the task, the collecting of information about requirements and the main constraints. The second is conceptual design, the search for a suitable combination of solution principles (Pahl, 1988), otherwise commonly called preliminary design - during which the central concept for the entire design is explored (Akin,1988). The third and fourth stages are concerned with detailed design. The third stage involves layout (embodiment) of the main elements of the design solution (determined from the decomposition suggested by the conceptual design decisions and resolution of the main constraints or component interactions). This is finally followed by the fourth stage which is detailed design of the individual components. All process models permit iteration between stages and most proponents of these models concede that the stages cannot easily be clearly separated.

> "The main phases of the design process cannot always be clearly

> delimited. Thus even a conceptual decision may require a scale drawing for the purpose of deciding on possible layouts. Conversely, the preliminary layout selected during the embodiment design phase may involve nothing more than rough sketches. Moreover, certain optimisations may be postponed until the detail design phase." (Pahl, op.cit., p.43)

In general, early works from the 1960s on design theory (the outcomes of conferences and workshops for example) contain the most detailed and prescriptive "scientific" models of the design process. These are presented alongside methods to be applied during the process which are described in the same vein. Good examples of this period are the works of Asimow (Asimow, op.cit.) and Gregory (Gregory, 1966).

1.2 Evaluation With Hindsight

This highly prescriptive approach failed to significantly influence design practice for a number of reasons. One reason is attributed to the weakness of the scientific basis on which notions of a scientific approach to design was founded. A second is concerned with misunderstanding and misappropriation of design methods themselves. Each of these reasons is discussed in turn below.

1.2.1 Problems with the concept of scientific method

In the late 1960s the movement to define design as a science was at its strongest. Design scientists acknowledged a fundamental distinction between design and science summarized by Simon as the concern in design with "how things ought to be" which he contrasted with science's primary concern with "how things are" (Simon, 1969). This distinction leads to fundamental differences in the kinds of reasoning which predominate in each case. In design synthesis predominates. In science analysis does so.

Design scientists looked to Popper's model of scientific method, that of conjecture and refutation (Popper, 1963), with the intention of adopting it and adapting it to arrive at a science of design. Conjecture, the process of producing a hypothesis in science, according to Popper's view, is formulated in such a way as to be falsifiable by application of theory, experiment and logical argument. In contrast, in the world of designing, it is argued that hypotheses are chosen and oriented towards success and justification giving a conjecture - analysis view of the design process. Thus, if Popper's view of scientific method is accepted and adopted, the view of design as science is problematical on this simple basis, namely on the significant divergence between the role and use of hypotheses produced in scientific research and those produced in designing.

A separate issue, but one which is more undermining to the transplanting of the conjecture and refutation view to design is the lively debate about the rational reconstructing of scientific progress which continues to occupy philosophers of science. A discussion of the competing theories of scientific discovery is beyond the scope of this work but what is pertinent here is that Popper's view was seen to have limitations as a model of scientific method among which is that it requires an external observation point from which a hypothesis can be evaluated from a "neutral" position. This is the weakest point of Popper's view i.e. the point which is most convincingly argued against; for the notion of a neutral position is widely attacked and held to be an impossibility. From the point of view of trying to establish design as science it is not a specific argument *against* Popper that caused the problem so much as that there *is* argument[1]. What is important for the development of ideas about design is that the "epistemological chaos over the concept of scientific method" (Cross, 1981) weakened the case for a design science which was based straight-forwardly on transplanting a view of scientific method to establish an equivalent view of design.

> "... the important lesson to be drawn seems to be as follows. Attempts to equate 'design' with 'science' must logically be predicated upon a concept of science that is epistemologically coherent and historically valid. The history of the twentieth century debate in the philosophy of science suggests that such a concept does not yet exist." (Cross, op.cit., p.198)

It is worth speculating here about what the design scientists were really looking for from science. Cross (op.cit.) holds that it was the apparent *values* of science rather than any directly transferrable method that was sought. The values of rationality were strongly desired as we have noted earlier. Escape from subjectivity was felt to be a worthy goal, and an attainable one. Ideals of neutrality and objectivity were in fact the strong motivations.

For the time being, at least, the quest for a design science based upon the shaky foundations of scientific method receded in importance. But what of the design methods, those procedures and techniques borrowed from other disciplines which were collected together and held out to designers and managers of design as a means of improving design practice?

---

[1]Actually in design the notion of absolutes against which to judge style, colour, form and so on is pandemic, think of the status of the classical orders in architecture, Pugin's influence on Victorian architecture based on his view on the (absolute) status of the gothic style, and just some examples from this century: the Modern Movement (Le Corbusier, et al.), the Purists (Ozenfant, et al.), and the basis of the work of the most prominent members of De Stijl and the Bauhaus.

1.2.2 Failure to change design practice

In advocating the widespread use of design methods, the intention was to move towards a rational, "scientifically" reasoned approach to designing, and therefore, implicitly, to move away from what were thought to be unscientific, intuitive, idiosyncratic ways of designing. These latter ways were deemed undesirable largely because they were not readily understood and were therefore difficult to explain or even describe in logical terms. They were thought unsuited for "scaling up" to the large, complex design challenges which have to be faced in a highly integrated, technological society.

At their best design methods were intended to permit communications that would otherwise be impossible, to increase the scale of action so that "what was before inevitable becomes now a conscious choice" (Jones, 1991). The fundamental reason for design methods failing to make a major impact was exactly because the deliberate intention was to change fundamentally the way design is carried out. With the benefit of more recent developments of ideas about design it is possible to see that this change was not effected for two reasons. In the first place design methods tend to address the "easy" parts of a problem - so they simply do not help much. "Each method begins with a first stage that is extremely difficult to do which has no description of how to do it which is intuitive." Thus wrote Jones in 1991 (op.cit.) about his 1970 text on design methods. Secondly, and more profoundly, the change in practice required amounted to a denial of the value of any reasoning which was non-scientific, or which could not be externalized. This, as will be seen later in this chapter, amounts to a requirement for change in human nature itself.

Practising designers did not radically alter in response to design methods. Rather, some of the methods, seen to be of use for particular stages of design, for particular classes of problems, were adopted and incorporated piece-meal into practice where they served a useful purpose. Interesting clues foreshadowing later development of ideas about design can be identified in the comments of some key exponents of design methods reflecting on their earlier work and the way in which it was received. A common criticism among them has been that design methods were appropriated by design methods researchers and led to an isolated field of research disconnected from the practice of design. Alexander in the 1971 preface to the paperback edition of his "Notes on the Synthesis of Form" first published in 1964 (Alexander, 1964) wrote

> "Since the book was published, a whole academic field has grown up around the idea of 'design methods' - and I have been hailed as one of the leading exponents of these so-called design methods ... I want to state, publicly, that I reject the whole idea of design methods as a subject of study, since I think it is absurd to separate the study of designing from the practice of design."

Jones writing in 1977, seven years after the publication of his design methods text, expressed a similar disappointment about the work in this area, setting his criticism in a wider context.

> "Instead of being the means by which professional practices in design and other fields could be despecialized and made more sensitive to human needs the new methods have become convenient tools for larger and more rigid planning and have also become the means of making design into a barren academic subject removed from life, from the lives of those for whose benefit it is supposed to exist, ourselves as consumers and users of industrial products." (Jones, 1991, p.31)

From the start, it appears, he had intended the methods to be used to *broaden* the range of what was considered. "Methodology should not be a fixed track to a fixed destination but a conversation about everything that could be made to happen" (Jones, 1970).

1.2.3 Contribution from the prescriptive era

What remain now of the ideas of this *prescriptive* era of design theory? Lifecycle models of the design process remain in use. Highly systematic approaches to design based on detailed descriptions of stages and steps are found most commonly in the area of engineering design (Pahl, 1988). Less rigid, but nevertheless disciplined approaches, where methods are grouped and are advocated as possible strategies for application within a framework of flexibly connected stages of design are also found for engineering design and industrial design applications. A good example of this approach, given by Cross (Cross, 1989), is shown in figure 2.1.

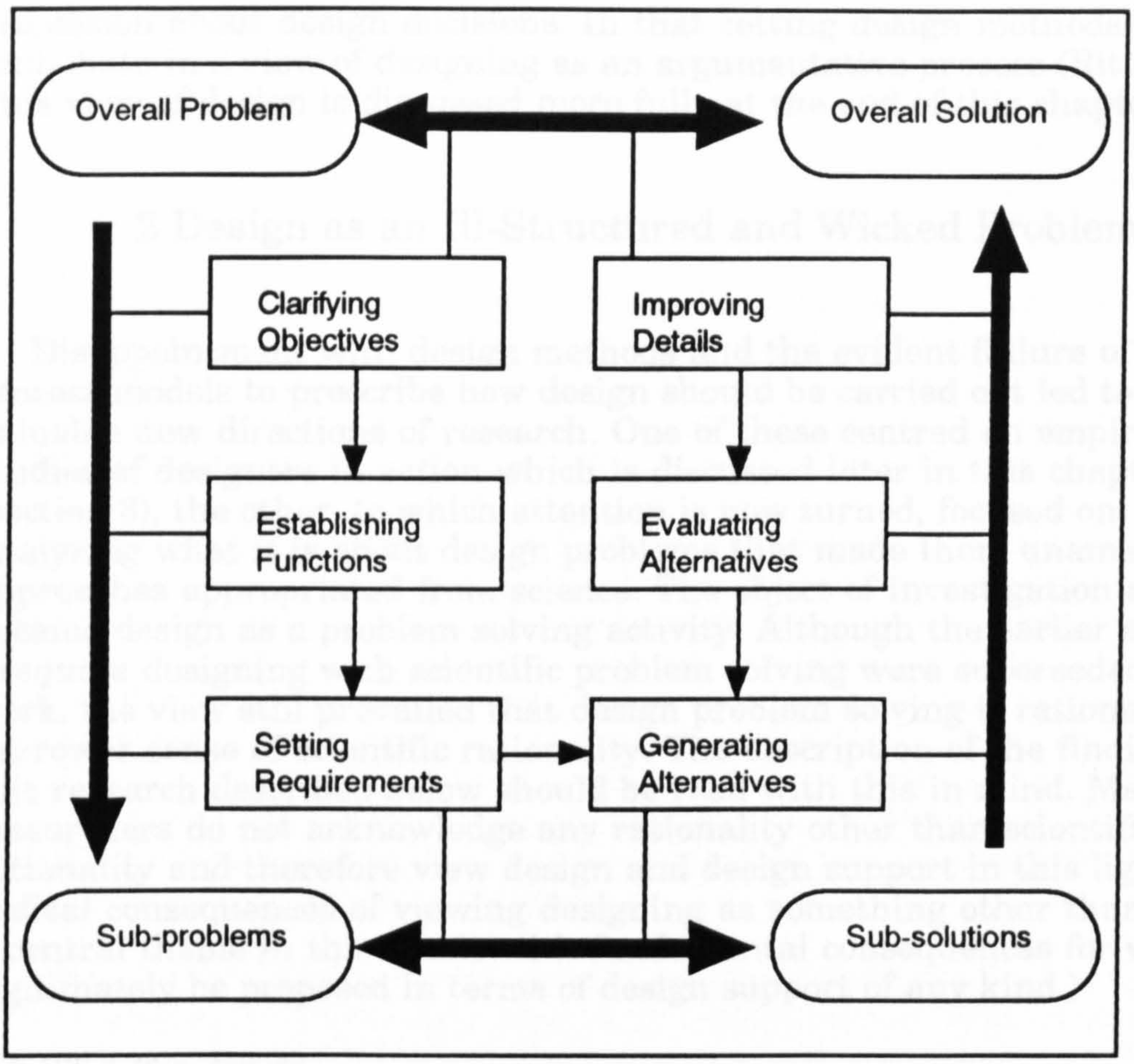


Figure 2.1 Design process as six stages within a symmetrical problem-solution model.

From Cross,N. Engineering Design Methods, Wiley, 1989, p.43.

In this model, design is viewed as six stages connected in a flexible way which allows for both decomposition of the design problem and reworking of stages of the design without rigidly prescribing either. Cross presents design methods which can be used within each of the six stages as strategies for tackling the stages to which they apply. The value and strengths of each method are emphasised so that designers can, if they wish, choose a method which will suit the characteristics of the problem at each stage.

It is worth noting that this particular model explicitly addresses two complementary aspects of design which Cross calls "understanding the problem" and "finding the solution". This idea will be explored fully later in this chapter. Design methods have also been retained for their value in investigation, in providing information which can be used to structure

discussion about design decisions. In that setting design methods contribute in a view of designing as an argumentative process (Rittel, 1972). This view of design is discussed more fully at the end of this chapter.

2 Design as an Ill-Structured and Wicked Problem

Disappointment with design methods and the evident failure of design process models to prescribe how design should be carried out led to two valuable new directions of research. One of these centred on empirical studies of designers in action which is discussed later in this chapter (section 3), the other, to which attention is now turned, focused on analyzing what it is about design problems that made them unamenable to approaches appropriated from science. The object of investigation now became design as a problem solving activity. Although the earlier attempts to *equate* designing with scientific problem solving were superseded by this work, the view still prevailed that design problem solving is rational in the narrower sense of scientific rationality. The description of the findings of this research described below should be read with this in mind. Many researchers do not acknowledge any rationality other than scientific rationality and therefore view design and design support in this light. (The radical consequences of viewing designing as something other than this is a central theme in this thesis with fundamental consequences for what can legitimately be proposed in terms of design support of any kind.)

2.1 The Findings

Design first had to be distinguished from logic and empirical science. This distinction was succinctly made by March (March, 1976) thus,

> "Science investigates extant forms. Design initiates novel forms. A scientific hypothesis is not the same thing as a design hypothesis. A logical proposition is not to be mistaken for a design proposal."

Well-defined or well-structured problems are considered to be those where the problem can be clearly defined, where the goal can be stated, and where attainment of the goal (synonymous with solving the problem and getting the correct answer) is achieved and can be shown to have been achieved by following a pre-defined sequence of steps or a procedure of some kind[2]. In well-defined problems, it is always clear when a solution has been

---

[2] Unless this definition is widened to include problems which at least can be shown to be achievable by following a sequence of steps (monotonically), then the definition hardly accounts for "mainstream" scientific problems let alone the challenge of designing. For instance, the creative, inspirational aspect of (new) scientific discovery is not adequately catered for in this description. Koestler (Koestler, 1964), for example, gives a number of accounts by scientists of how they came to develop new theories. These do not fit

obtained since the criteria for testing the solution are clear (Newell, 1967). Problems like these are often loosely described as ones that can be solved on a "scientific" basis. Prescribing a procedural design process model which is entirely systematic is effectively an attempt to put design problem solving onto a scientific basis, and as a consequence to classify design in a certain way.

However, design, except that of the most trivial and constrained kind, is not the finding of a solution to a well-defined, well-structured problem. Design problems are *at least* ill-defined ones (Newell, op.cit.). That is, they are problems where even given some general outline statement of what the problem is, the means of solving it, and the "goal" is not given at the outset.

Design problems are *very* ill-defined ones. Their context is broad. They rank alongside the most complex problems faced by human decision makers (Rittel, 1973). Very ill-defined problems have been dubbed *wicked* problems to describe the challenge they present by contrasting them with tamer, "straightforward" problems. The description of the intrinsic nature of design problems which follows emerged in the early 1970s. The research which led to this was initially prompted by failure, in a number of fields, of procedures suited for application to well-structured problems. The characteristics of wicked problems are described below in terms of how they are manifested in design problems.

Viewed as wicked problems, most design problems can be seen to have certain properties. In the first place they cannot be described in detail in the form of a problem specification. To attempt to do so would be to start to impose ideas about the design solution upon the problem statement. A design problem is initially poorly specified, the goals are vague, the problem context is complex. What is relevant in terms of constraints are not known (or knowable) initially. Since the goal is not clear, it follows that it is not obvious when the design is finished i.e. when work on it should stop. There are always possibilities of doing better with more effort or by considering something else i.e. a wider context, or more alternatives. (Once again this aspect of designing will be seen as a recurring theme in this thesis, to be explored in more detail with fundamental consequences for design support of any kind.)

It is not possible to exhaustively describe the set of possible designs. A finished design will be adequate or inadequate, appropriate or inappropriate, good or bad, but not correct or false. There is no clear demarcation of the consequences of a design. Seen as a solution, there is no way of testing the design exhaustively. In many design domains, the possibility of trying out a design to see if it works as a solution in situ i.e. "in real life" is not a possibility. For example a bridge, a motorway, a power

---

this model too well, he quotes Polya for example thus, "when you have satisfied yourself that the theorem is true, you start proving it"(p.118).

network or a building cannot be built to see if it satisfies the need identified and to see that is does not have unpredicted negative effects. Although this is the case the designer is not accorded the privilege to be wrong. Two design problems may exhibit similarities, and yet there is no guarantee that what is different about their individual contexts will not have some overriding effect on what it is appropriate to consider and to propose as a solution.

An initial statement of a design problem, however poorly or vaguely described is always itself a statement of a proposed solution to a higher level problem which is broader and more general. It is alleged that some design failures are caused because the initial problem statement has imposed a starting point for design decision making which is at too low a level (Petroski, 1985). This particular "recipe for disaster" is a common cause of failure in information systems design, for example. In this design domain, there is almost a tradition of giving the designer a “starting point” at the level of designing a computer system to support some existing data intensive procedures in order to make them more efficient, less labour intensive, or to improve the quality of data. It was only really at the start of the 1990s that it began to be generally realized that design of the organization itself and its procedures (based on what information systems technology can do) is the effective level at which to address information management and hence information systems design (Hammer, 1990).

2.2 Contribution of the Descriptive Era

There is no doubt that research which has focused on the intrinsic nature of design problems, and which has led to the acknowledgement of the properties of design problems outlined above, has made a significant contribution to our understanding of what designing is about. Jones' puzzling statement that all his design methods required an initial intuitive step that was extremely difficult to do (see section 1 of this chapter) starts to make sense if we begin to differentiate between setting a problem and solving it. Many of the design methods can now be seen to be applicable to the solving of well-defined problems. This is just one aspect of designing. A designer having arrived at a point where he or she can see the value of solving a particular well-defined problem which will contribute to the overall design can make use of appropriate design methods within a limited context that he has identified to solve a problem, to choose between alternatives, or to help inform the debate about possibilities. The "difficult" part of design however, that which the designers handles as a result of his training and experience, is the problem setting. In other words, the hard part of designing is the determination of what problem or set of well-defined sub-problems are to be solved using design methods or by other means. This aspect of designing, the problem setting, will be the central concern in the remaining sections of this chapter.

Before moving on to look at what observation of designers in action has

revealed it is useful to note that a valuable result of investigating the nature of design problems has been the classification of design activities. Design clearly covers a wide spectrum of activities. At one extreme is found the most inventive, highly creative kind of designing the sort which is accompanied by cries of "Eureka!". At the other end of the spectrum is found very routine types of work where minor alterations to a previously defined form of solution constitutes the design activity.

The boundaries between types of design have not been rigidly defined but some broad classification is widely accepted. Typically, three categories are identified. The first is *original* design, innovative in nature, in which the principle of the solution is novel for the design task. The second is *adaptive* design in which a known solution principle is adapted to a different design task. The main design is achieved in this way although component parts of the design may be original designs. The third is *routine* or variant design in which variations to the values of parameters or some rearrangement of the component parts of a previous design are made (Pahl, 1988).

3 What Designers Actually Do

Many of the empirical studies of human designers in action investigate designers who are either architects or are from disciplines such as environmental design which are closely related to architecture. It is not clear why this should be so predominantly the case, however, some factors which may be salient are offered here. Firstly, it appears that conscious reflection about the act of designing is encouraged in architectural and some industrial design disciplines to a far greater extent than in engineering fields. The education of architects in design studio environments provides rich opportunities for reflection and discussion about design processes as well as for observation of them (Schön 1985). This environment does not routinely form part of an engineer's educational experience. Secondly, the products of architectural design are exposed to public view and public scrutiny in a way that much of engineering design is not. People know what architects do on the whole, and in the United Kingdom at least architecture invokes strong views from a complete cross-section of society (Windsor, 1989; Kolb, 1990 (p.128)).

All designers affect peoples' lives, but architecture does so in a very direct and easily attributable way. The crisis in architecture over the problems of its products in the 1960s may be a reason why architects have become introspective and why others have focused so much attention on how they do what they do. Finally, architecture has perhaps been of particular interest because it is the design discipline that stands at the crossroads of art, science and technology. In architecture the design process has been described as the point where ingenuity and art meet (Le Corbusier, 1927) and the product of this kind of designing has been described rather poetically, but in the same vein, by Brancusi as inhabited

sculpture (Copplestone, 1991).

None of these factors implies that the study of architects in action is an unsuitable way of finding out about what designers (more generally) do. It is clear, however, that in many design disciplines aesthetics (at least visual ones) or a focus on design for consumption play a more restricted role than they do in architecture. The remainder of this section is occupied by a review of studies of what designers actually do in practice. Each of the investigations contributes a perspective on some aspect of designers' behaviour.

3.1 The Solution-Centred Aspect of Designing

An experiment (Lawson, 1990) to investigate and compare the approaches to design problems taken by designers and scientists concluded that designers proceed by trying out solutions whereas those with a scientific background focus their efforts on studying the problem directly. Participants were given partial information about acceptable solutions and their submitted designs were either accepted or rejected on the basis of the full requirements (rules) some of which were not disclosed initially. The non-designers, i.e. the scientifically trained participants, appeared to submit designs in order to discover the hidden rules - their strategy being to focus on the problem. The designers however produced solutions which satisfied the requirements given and then revised their solutions until they produced something satisfactory - their strategy thus appeared to be solution-centred. Scientists analyze the problem *in order to synthesize* a solution in contrast to the designers' approach which is to learn more about the problem *by synthesising* solutions. According to Lawson, these different ways of approaching design problems are acquired through different experiences and values in education, training and professional practice. Designers learn by example and practice, and are judged by the design solutions they produce rather than by the methods by which they arrive at those solutions.

3.2 The Roles of "Organizing Principles" and "Design Generators"

Rowe (Rowe, 1987) presents three case studies of designers in action. Designers recorded the sequence of their design ideas and kept ordered records of their working notes and drawings. At intervals during the design projects the designers were interviewed. During interviews they described their work so far and answered questions for clarification. Rowe describes the designers as each taking stock of their design problems seemingly from a preoccupying orientation. The orientation appears to set to work something which he terms "organizing principles". The designer takes a theme which may be something abstract like "to build something which visibly reveals its flexibility" (like Richard Roger's account of his conception about the Pompidou Centre in Paris) or which may be drawn from pursuing the logical consequences of some analogy. The approach of

pursuing a concept or theme consequently furnishes organizing principles which allow the designer to get a grasp on the design problem, to explore its possibilities (cf. Akin's scenarios in section 3.4 below). The organizing principles have a direct bearing on the emergence of design proposals and are thus seen by Rowe as an enabling force which both empowers the designer with the ability to do the designing and which influences the way he proceeds with it. Rowe observed that the ideas and references from elsewhere which the designers in his studies brought to bear on design problems were "particularly evident during the early stages of the projects, as the designers searched for concepts around which to construct frameworks for reinterpreting the design problem" (op.cit. p.37). The role of these concepts appears to be to provide insight and direction for further exploration.

Rowe describes the design process in terms of the designer moving back and forth between the design problem as given and the tentative proposals the designer has in mind, between exploration and evaluation. Progress is not made linearly, rather as episodes during which various aspects of the problem are investigated.

> "Within the episodic structure of the process, the problem, as perceived by the designer, tends to fluctuate from being rather nebulous to being more specific and well-defined." (op.cit., p.35)

As work proceeds investigations begin to converge, the design activity takes on a direction. However designers speculate by persisting in following the consequences of a particular orientation (given by the organizing principles) beyond the point where it seems realistically tenable. They then may return to an earlier point of departure. Sometimes they exhibit an apparently more systematic (conscious) exploration of variations on an organizing principle and evaluate their relative worth.

Rowe concludes that the ideas and references that a designer *brings* to a design problem figure more prominently in shaping his decision making , at least during the early stages of design, than do the particulars of the given design problem itself. Initial design problems, as formulated through the organizing principles, and external references have a sustained influence on the emerging design. Thus normative reasoning is being brought to bear, influencing the design in a significant way. This leads to the conclusion that investigation of designing based solely on an investigation of the problem solving process will not be adequate to explain what is going on.

Earlier work (Darke, 1979) in which designers (architects) were questioned about housing design projects after their completion seems to indicate that a single idea or related group of concepts "form a starting point ... a way in to the problem". These concepts fulfil a role as initial *generators* for design solutions. A designer "does not start by listing all the

constraints. Any particular primary generator may be capable of justification on rational grounds, but at the point where it enters the design process it is usually more an article of faith". The initial generator is a constraint imposed by the designer to give him or her a way of reducing the variety of potential solutions. Designers appear to fix on a particular objective or small group of objectives, ones which are based on things which they, as individuals or as a professional group, value highly, which are self-imposed. They do this "for reasons that rest on their subjective judgement rather than being reduced to a process of logic" (Darke, op.cit.). Darke proposed that the conjecture-analysis model of the design process (the origin of which was outlined in section 2.1 above) be modified to acknowledge these findings. She proposes a "generator - conjecture - analysis" view as the basis for elaboration through further research.

Darke's studies also show that designers tend not to separate out aspects of the design that satisfy different requirements. Direct mapping of individual constraints or objectives onto identifiable components of the design does not occur. Process and product are seen holistically. The various requirements are seen to be facets of a single problem or design challenge and they are solved in an integrated way. On this basis analysis-synthesis models (like that of Asimow described in section 2.1) of design are dismissed.

An experienced designer knows how much of what is initially presented or which subsequently becomes relevant during the design task can be called into question (Lawson, op.cit.). "Creatively uncovering the range of his problem is one of the designer's most important skills" (Rowe, op.cit.). A designer is expected to contribute something to the task for which he has been commissioned and which is usually initially ill-defined. He expects his design activity to lead him to reformulate the design problem. When the designer evaluates the product of his speculative excursions he is both checking for the expected behaviour or outcomes that lead him in this direction in the first place and also noticing unexpected outcomes which gives him new material to inform his view of design problem and possibilities for tackling it. In this respect design can be viewed as an "exploration process, what is relevant only manifests itself as the design proceeds and varies with the decisions taken" (Gero, 1990, p.29).

3.3 Schön's Views on an Epistemology of Practice

Schön (Schön, 1983, 1985, 1987, 1992) examined what individuals from a number of professions including architecture, engineering and planning actually do by analyzing episodes in which experienced professionals attempted to help junior ones to learn from practical experience. Schön's work is based upon the direct recognition that competent designers (in common with other practicing professionals) are able to do their jobs well but are not able to define what they do. They demonstrate a kind of "knowing-in-practice" which is tacit, they operate intuitively (Ryle, 1949) in

interpreting a situation. Schön's studies are directed towards a better understanding of what he calls the epistemology of practice and opposes the tendency which dismisses this kind of knowing as something disturbing, to be overcome or avoided because it cannot be made sense of from a technically rational perspective. That is from the restrictive view of professional practice which sees problem solving as solely the rigorous application of scientific theories and techniques. Schön's view is that this kind of problem solving is a part of professional competence but it is set within a broader context which is constructed by the designer to make sense of the complex, unstable, uncertain situations which he is presented with when he is given a design brief.

Schön describes professional practice in general as consisting of both problem setting and problem solving.

> "When we set the problem, we select what we will treat as the 'things' of the situation, we set the boundaries of our attention to it, and we impose upon it a coherence which allows us to say what is wrong and in what directions the situation needs to be changed. Problem setting is a process in which, interactively, we name the things to which we will attend and frame the context in which we will attend to them." (Schön, 1983, p.40).

Problem setting is the process of defining what decisions have to be made, what objectives are to be accomplished, and how to set about achieving them. In other words, problem setting is how designers cope with wicked problems.

Schön presents his empirical studies to support the idea that a major characteristic of experienced designers is their ability to think in a particular way about what they are doing while they are designing. Schön calls this "reflection in action". It is a key component of professional practice because it is through reflecting in action that, faced with something out of the ordinary, a new, a unique challenge, the designer can, as it were, rise above his initial attempts at tackling a problem and can criticize his own approach - his initial understanding of what was presented - and can construct a new formulation of the problem. That is to say, he can make sense of it in a different way. Acting as a enquirer, the designer, as he designs, is open to the discovery of things which are incongruent with the initial problem as he has set it for himself to tackle. He can be surprised by what he finds, he can reflect on the experience and reframe the problem in the light of this.

Described in this way, the designer can be seen to be holding a kind of dialogue with the problem situation, there is a sense in which he interrogates it. The designer's initial problem framing allows him to make progress, to explore and evaluate the implications of pursuing his initial ideas. The situation "talks-back" to him, confirming his commitment to the

chosen approach or surprising him with initially unseen implications. He reflects on this, he experiments with problem reframing, he receives new and possibly unexpected talk-back from the situation which redirects or further focuses his efforts as he re-appreciates, reinvents, explores and commits himself to design decisions.

3.4 Problem Structuring and Restructuring

Akin (Akin, 1986, 1988) has studied what he terms the problem structuring behaviour of designers and non-designers in a number of experiments. It appears that designers and non-designers perform similarly when given well-defined problems which require the satisfaction of pre-specified constraints. Differences are revealed when ill-defined problems are to be tackled. Akin concludes that this contrast in performance points towards a critical component of designers' expertise. A designer applies principles and methods which are needed to solve well-defined problems, however the essential difference for designers is that the designer has to find ways of bringing these principles to bear on ill-defined problems in a productive way. Akin calls this the task of problem structuring.

> "As the architect develops solutions or partial solutions that begin to meet some of the requirements of the initial problem description, comprehensive evaluations of these solutions are performed. Next, the architect invariably alters the structure of the problem in ways which lead him to more successful results. A common form this restructuring takes is the addition or deletion of problem constraints or solution parts ... from the initial problem description." (op.cit., p.179)

Detection of conflicts is a major influence on problem restructuring particularly when a design is evolving. Other causes of problem restructuring are at work however, for example, when the implications of a number of ideas are being considered before selection of a particular approach to the solution. Restructuring is recursively applied as design progresses, sub-problems are identified and in solving these, new requirements and constraints between them emerge. In his study of architects Akin identifies several strategies that they commonly use to structure or restructure design problems. They use conceptual constructs, which he terms scenarios, for organizational purposes. These play a particularly strong role in the evaluation of a design; by trying out a number of scenarios, designers get to explore diverse possibilities before committing themselves, and in doing so they develop a more comprehensive understanding of the consequences of design choices. He also identifies that designers use a process of formally considering different physical alternatives to explore alternative problem structures, these he terms prototypes. These too contribute to the creation of order, and thus to the structuring of problems. Designers consider alternatives before

selecting a particular problem structure and they evaluate the extent to which the alternatives satisfy the overall goals of the design. Scenarios are abstract templates which link functions in relationships which can be adapted to different physical constraints. Non-designers rely more heavily on specific templates (physical prototypes) from their personal experience which are less adaptable. Designers consider a number of alternative problem structures.

> "Different scenarios often enable the designer to study solutions which are of completely different types. This leads to the consideration of diverse possibilities and a more comprehensive understanding of the ramifications of design choices." (op.cit., p.181)

Designers tend to explore the problem, to interpret the situation presented, more broadly than non-designers. This breadth is demonstrated in two ways. Firstly, even though a particular problem structuring seems promising, a designer will tend to consider alternatives and the problem restructuring consequent upon them. Secondly, designers consider in some detail the implications of ideas which seem a priori to be unpromising. They avoid adopting a solution, that is, pursuing a particular problem structuring until a number of strong alternatives have been considered. In Akin's studies the result of this behaviour was that designers appeared to generate, and have to resolve, more high level (global) conflicts than non-designers who do not face the same difficulties because they do not restructure the problem so much. Evaluation takes place as solutions are explored to see how well they satisfy the overall design goals. Evaluation leads to modification of requirements.

> "As solutions or partial solutions are developed architects evaluate the degree to which these satisfy the overall goals of their designs. If they find that certain requirements are restricting the emergence of 'good' solution ideas, then these requirements become candidates for being discarded. If some desired solutions suggest requirements not yet identified in the program, these become addenda to the requirement list. If new scenarios are suggested by the earlier problem structures, then an entirely new set of requirements are developed and a new agenda of explorations is identified. Thus, evaluation of earlier design steps becomes the key for finding successful future steps for the design process." (op.cit., p.181)

Akin's experiments lead to findings which clearly closely link problem restructuring to evaluation of previous problem structurings or attempts at structuring. This behaviour is not restricted to architects as is shown in the case study which forms the second part of this thesis (see in particular chapter 7 sections 2,3, and 4).

3.5 Opportunistic Use of Resources

Visser (Visser, 1991, 1992) has observed the actual activities that occupy designers in order to investigate the way in which they organize their activity, what strategies they use and what are their problem solving processes. Her observations were of designers in the fields of mechanical engineering (machine controller design), software engineering and the design of composite structures. She found that the structure of the designers' actual activity (as directly observed) differed from the description of the activity structure given by the designers during interviews. A designer tends to describe his actions in terms of following, fairly closely, a hierarchically structured plan. In practice, however, the designer tends to proceed much more opportunistically. The designer "deviates" from planned action whenever circumstances are favourable. He does not always "resume" a plan having deviated from it, although he may do so. This behaviour is influenced by how constrained is his design brief.

Visser has analyzed deviation actions and finds the following causes of deviation. Designers take advantage of information as it becomes available; they carry out processing tasks that they perceive to be similar or closely related in some way; and they sometimes "drift" to focus their attention on another aspect of the design when they are experiencing difficulty with the current focus. Design activity appears to be strongly opportunistically organized and not hierarchically organized as is commonly supposed (even by designers themselves). (There is a body of related work, for example studies of planning, which supports these findings (e.g. Suchman, 1987). These studies in other areas are beyond the scope of this thesis.) Visser's work adds evidence to the idea that designers focus attention on elements of the design but they abandon and resume attending to these elements according to preference influenced by the information which becomes available and the semantic relations between the elements of the design. The designer uses his resources effectively in terms of both the time spent and the effort expended.

3.6 Impact of Empirical Studies

Before moving on to the final section of this chapter it is worth making two observations about the investigations of what designers actually do as reported in this section. Firstly, whilst different investigators have coined different terminology for the phenomena they have identified in analyzing their observations and each researcher has a slightly different primary focus of attention it is clear that the findings overlap to some extent. There is no contradiction which emerges, rather where each piece of investigation overlaps another it tends to reinforce the arguments presented in each case. Secondly, the findings reported here are all largely disregarded, either consciously or through ignorance, by those involved in building knowledge based systems for design whose "natural" home is among the A.I. community.

4 The Essential Nature of Design

What then is the outcome of the research over the last thirty years into the nature of design? The focus of attention has shifted during this period from the design process, to design as a problem, and finally to the designers themselves. The results of the separate research themes have influenced the others and has contributed to the overall understanding of design that has emerged from it.

Some researchers view design, and in particular engineering design, as a kind of *problem solving*. Viewed this way, designers are seen to make use of a body of knowledge, associated with the domain within which the design falls, suitably structured for the purpose of designing particular kinds of things. These designers take a disciplined approach to the development of the design. Their approach is structured in a way which they have found, through experience, to be effective. As design proceeds, the designer evaluates what he produces. Unsatisfactory aspects of evolving solutions provide information leading to changes of strategy and to improvements in the developing design and ultimately to progression towards completion. So, as design progresses, the designer acquires information which affects the way in which further progress is made or attempted.

The view that design is "just" another kind of problem solving activity is not a universally acknowledged one. Evidence from diverse sources shows that design is an exploratory process in which designers' actions may be viewed as a means of *understanding a design situation*. The examples of investigations into what designers actually do described in section 2.3 above support this view. Contributions to it are to be found in Rowe's investigations of designers' organizing principles at work within an episodically structured process, Schön's description of professional practice as problem setting and problem solving, and Lawson's experiments showing how designers use trial solutions to learn more about the design tasks with which they have been presented. From this *solution oriented* perspective, as well as from the problem-solving one, designers are seen to approach a design task with experience and with knowledge, acquired and structured in a useful way through practice. They apply themselves to the design situation as they perceive it initially. They explore the situation by trying out solution ideas on it, as we see from Akin's findings. As they proceed with this - the design task - and as they take decisions, what is relevant becomes apparent to them; they come to understand more about what they can do. The presuppositions of the designers are viewed as a means of allowing them to get a grasp on the situation as a whole in the first place, as Darke has shown, and through the particular professional perspective that they have, to begin to fill in the detail. As the detail develops, the designer increases his appreciation of the situation as a whole and revises the design to accord with the new understanding.

Designers pre-structure design problems to make them tractable and thus to solve them. They do this by bringing ideas, making analogies, and using metaphors from their experience. They use organizing principles drawn from, or distilled from, the professional group or school of designers to which they belong and in which they practice.

Designers create requirements and design constraints which are added to and modify the initial specification which forms their design brief. Designers set the problems they solve. The problem setting is the structuring of the problem which renders it solvable by "rational" problem solving methods. Designers focus on solutions rather than problems. Designers shift attention from consideration of the whole to consideration of the parts that constitute the whole. Consideration of the parts influences their ideas about the whole and so on in a cyclical fashion. Viewed in this way designing can be thought of in terms of a conversation, a dialogue in which the design (situation) and the designer are the participants, each changing the other as the design unfolds. Designing is concerned with interpreting a situation, the process of designing is the process of understanding the design situation.

If this richer picture of design is accepted, design is inevitably "relocated" from the natural to the human sciences. This shift does not rule out a role for rational, science-based problem solving methods. These play an important role as activities in design, but the development of ideas about designing over the last thirty years has set them in place, in a wider context of problem setting and problem solving, and the wider context still of the primary generators, enabling prejudices, the organizing principles that the designer brings to bear from his professional background in setting the design problem itself.

When designing is viewed in terms of coming to an understanding and interpreting design situations interest turns to the human sciences and those ideas within it that are concerned with what it is for humans to interpret and understand the situations with which they are faced. This tends to focus attention particularly on phenomenology and the modern hermeneuticists. This recourse to the human sciences should not be viewed as a rejection of early theories about designing. The design methods movement, the investigation of the characteristics of design problems and the studies of what designers actually do have all contributed to the understanding of design which has now been reached. The human sciences make a contribution, adding to this understanding in new ways, by giving new insights into the essential nature of designing as a human activity.

4.1 Designing Expressed Hermeneutically

How does a designer come to understand a design situation? How does he identify, explore and interpret the possibilities which it presents to him? How does he come to appreciate what is important, what he must pay attention to, what he can call into question? How does he arrive at a design solution that can be justified in its setting?

Designing is an exploratory process. A designer in perceiving a design situation as a design situation at all is perceiving from a point of view. He does not wholly control the field of view which sets the context of the design situation. However the design task, as the designer sees it, presents him with room for manoeuvre, it presents him with possibilities (Merleau-Ponty, 1962). He comes to the situation with professional experience and with knowledge, acquired and structured through that experience and applies himself to the design situation as he perceives it initially, incompletely constituted. He explores the situation and as he proceeds with the design task and as he takes decisions what is relevant from the situation becomes apparent to him.

Designers' experience enables design. Designers' do not perceive the design situation from an arbitrary perspective. The perceived situation is a creation of the designer, created through the intentionality which he brings to bear (Merleau-Ponty, op.cit.). The professional choices made by a designer are part of a process which is inherently dialectical. They are all about choosing and deciding in favour of some actions and against others (Gadamer, 1981) on the basis of professional judgements competently executed. The presuppositions of a designer allow him to get a grasp on the situation as a whole and through this perspective to construe the details, from the recognition of the nature of the details he increases his appreciation of the situation as a whole (Ricoeur, 1981). Designers' horizons are not fixed. Although he makes a start from a particular perspective with specific given constituents, the designer is able to see beyond this stand-point, to move the horizons (Gadamer, 1975).

Designers are solution oriented. For the designer, problem understanding is inextricably linked with generation and exploration of solutions. To justify a design, to make sense of it, it must be compared alongside its alternatives. Designs are selected or proposed on the basis of their differences from other designs in the continuum of possibilities, not by the their merit in relation to absolute values.

What is called the design task? The design task is what the designer perceives himself faced with. Competent designers do not approach a new task from an arbitrary perspective or an uninformed viewpoint. The "prejudices" which the designer brings to a situation are an essential component of what enables him to see it the way he does as a situation in which he is able to practice his skill, and to make professional judgements.

Designers and designs interact. As the designer explores solution alternatives, the designs that are suggested themselves bring into focus new spaces of related information (Fischer, 1991) which are not determined a priori. Schön has described this interaction between the designer and the design alternatives as stimulating a reflective conversation (Schön, 1985, 1992) in which the designer comes to explore the situation with which he is faced. The avenues which he explores and the problems which he tackles in considering design possibilities are not simply left behind him as he goes along but become part of his experience and affect how he proceeds (Gadamer, 1981).

The hermeneutical view of the design process can be stated as follows. A designer who is trying to understand a design situation is, as a matter of course, performing an act of projecting. His projections are possible interpretations which allow him to grasp the challenge presented by the situation as a whole as soon as some initial sense emerges. The projections act as a framework for problem solving. The emerging sense he makes from the design situation comes about through the expectations he brings to the situation about the nature of the task. The working out of this initial projection, which is constantly revised according to what emerges as he enters into, as he explores the design situation, is how he understands it. Every revision of the projected solution is capable itself of invoking new projections. Rival projections can emerge side by side until it becomes clearer to what the totality of the situation and the design solutions amount. Interpretation of the situation begins with pre-conceptions that are replaced by more suitable ones. This constant process of making new projections is the movement of understanding and interpretation which constitutes the design task (paraphrasing Gadamer, 1975, p.236).

4.2 Some Research Challenges

Just over twenty years ago Rittel (Rittel, 1973) proposed that design be viewed as an argumentative process where designing consists of the "counterplay of raising issues and dealing with them which in turn raises new issues and so on and so on" (op.cit., p. 320). From this point of view the idea that design should proceed by the designer consulting the client to, as it were, "understand the problem", then going away, solving the problem, and then presenting the client with the solution is not a credible way of proceeding. At every step in the design, the designer is making judgements about issues as they arise. Rittel makes the case that clients must be "accomplices" in generating the design solution. An argumentative view of the design process requires that the statements made as the design proceeds are "systematically challenged in order to expose them to the viewpoints of the different sides" (op.cit., p.321). This process raises factual questions and questions about "what ought to be the case". Rittel sees a place for the design methods of the early days of design theory as tools to support or attack a point of view, i.e. as a resource for argumentation.

Without recourse to the language of hermeneutics, but in a way which is compatible with it, and which predates its appearance as a way of looking at design, Rittel has neatly described the design process as it is now understood thus:

> "one (insight is) ... that the design process is not considered to be a sequence of activities that are pretty well defined and that are carried through one after the other, like 'understand the problem, collect information, analyze information, synthesis, decide' and so on; and another being the insight that you cannot understand the problem without having a concept of the solution in mind; and that you cannot gather information meaningfully unless you have understood the problem but that you cannot understand the problem without information about it." (op.cit., p.321)

Rittel's specific challenge to the design methodologists is to focus research on developing and refining an argumentative model of the design process. (The work led by Fischer described in chapter 3 directly addresses Rittel's challenge to work on practical procedures for supporting design based on an argumentative model.) However, he proposes a wider programme of research suggesting that attention should be centred on how the designer explores the design situations; what directs him to ask certain questions, what prompts him to attend to generating information about certain aspects of the design problem; and how he arrives at judgements. The first part of the case study presented in part 2 of this thesis (chapters 6 and 7) is centred upon investigations in this area for a particular design domain. More specifically,the work reported there is concerned with how design proposals evolve from the exploration of solution possibilities; how the design proposals relate to the judgements made on the basis of the designer's assessment of the issues relevant to the design situation; and the part evaluation of emerging alternative proposals plays in furthering the design solution and in defending it under peer scrutiny.

CHAPTER 3

Using Knowledge Based Systems for Designing - A Review of Three Approaches

"The intrinsic worth of an individual exists only for him, and not for me; I can only get as far as his outward actions, and to him I am nothing more than an outer appearance, an absurd set of premises; premises which I do not even choose to be."

Simone De Beauvoir (The Blood of Others)

In this chapter three approaches to knowledge based system support of design are described. In each case the work described centres on a particular system which has been developed as part of a larger research programme. The particular systems described are :

a knowledge based system to design the mechanical components of air cylinders, known as AIR-CYL (the language in which it is written, DSPL, is also described), this has been developed as part of a programme of research into the notion that routine design is a generic task;

a system to support a designer of mechanical products, known as EDS, which forms the core system in an investigation into supporting design viewed as process which is essentially exploratory in nature;

and finally, a system to aid design of kitchen floor plan layouts, known as Janus, which provides domain specific building blocks and other knowledge based support for the designer, this forms part of a programme of work which pursues the idea of supporting human computer interaction by using (problem) domain terms, objects and operations, design is viewed as an argumentative process (cf. the ideas of Rittel chapter 2 section 4.2).

What these systems have in common is that each plays a role in supporting programmes of research into how knowledge based systems can be used to support designing. However, what they have actually been constructed to explore differs greatly, according to the view of design and the broader research purposes of those who have built them.

Figure 3.1 shows how the three systems described in this chapter stand in relation to one another and in relation to different perspectives on what support for design should constitute.

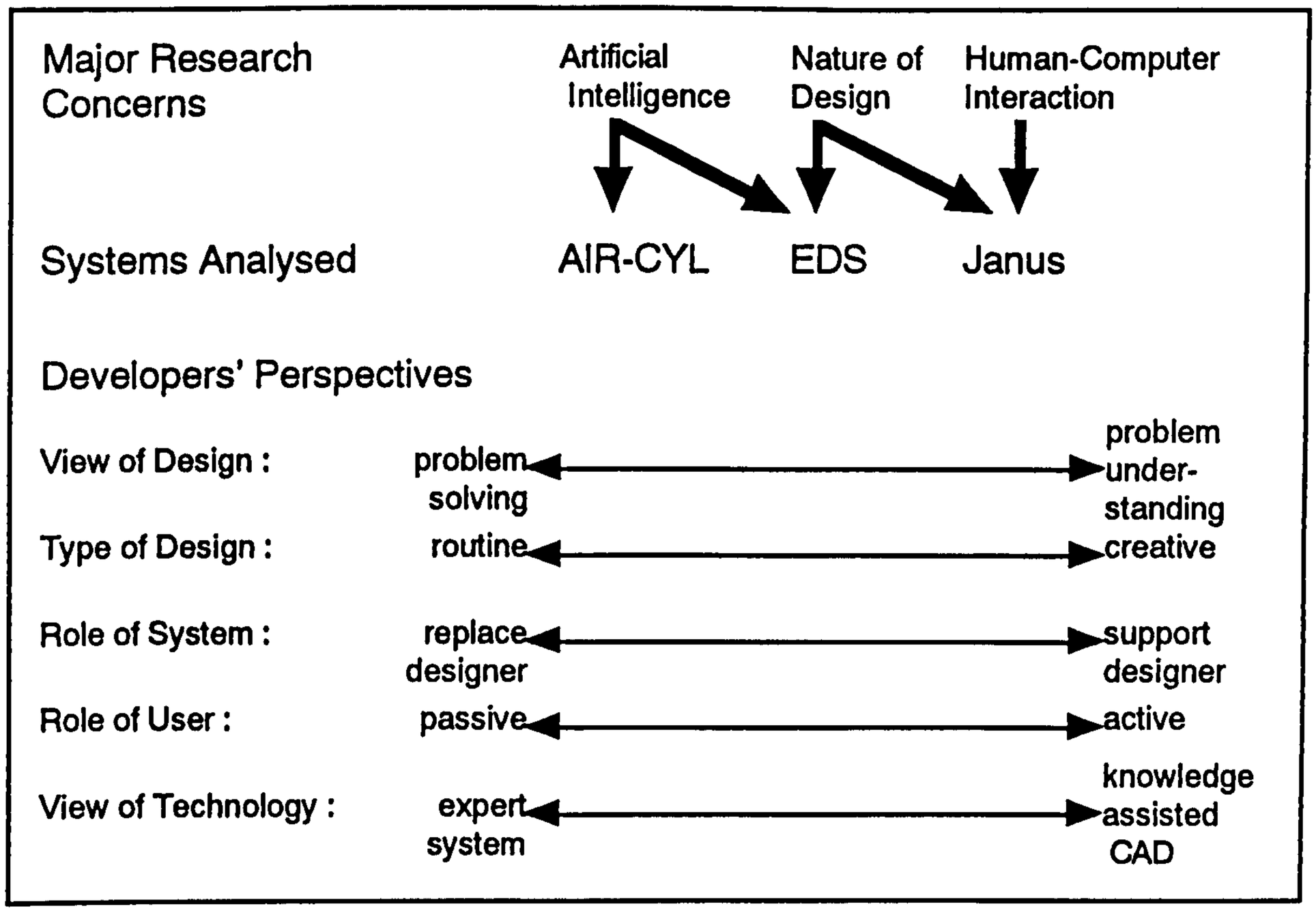


Figure 3.1 AIR-CYL, EDS and Janus in the spectrum of knowledge based systems for design.

The main reason for choosing to review these systems is that, between them, they cover the spectrum of concerns very well. The further reasons why each individual system has been chosen for detailed description in this chapter are as follows.

The first system described (in section 1 below), AIR-CYL, is a representative example of a product of the A.I. community's research into knowledge based systems for design. There are three main reasons why it has been selected. Firstly, it is an example of a second generation expert system which has been developed in a research group in which ideas about the relationship between the explicit representation of knowledge set within the context of the task structure and the ability to provide intelligently structured dialogue (including explanations) capabilities plays an important role. In this respect, at least, their work accords with some of the research into competent systems to which this thesis makes a contribution. Secondly, AIR-CYL is typical of work on systems to automate design which is going on within the A.I. community in terms of the attitude taken towards human designers both during development of the system and as users of it. Its validity in these respects holds despite the fact that the design task which AIR-CYL tackles is modest in comparison with some other systems. Thirdly, it has been chosen for the purely practical reason that it is very thoroughly described in publications. Unusually, for expert systems applications, its authors have laid their system open to detailed scrutiny by the wider research community.

EDS has been chosen for two main reasons. The first is that it tackles uncompromisingly the idea that design is an ill-structured and wicked problem. In this respect it differs from much of the work produced by the A.I. community. The system does not attempt to do the designing, i.e. to replace the designer, but it does attempt to do some of the supporting inferencing for the designer. The second reason for its choice, related to the first, is that it focuses on the most difficult part of the design process, namely the earliest stages of problem formulation and the conceptual stage of design. The case study of design which is reported in the second part of this thesis is concerned with this same stage of design, the preliminary stage. This tends to be the least structured part of designing when the concerns of the designer with problem understanding are at their most prominent.

The third system, Janus, comes from a community of researchers whose central concerns lie with issues of human-computer interaction and who explicitly express a desire to move away from attempting to produce artificial (intelligent) designers. It is of central interest to these researchers to find out about, and to support humans in, the way they actually work. Although their central concerns are with redressing the alleged imbalance towards the computer system component in human-computer interaction (HCI) and their fellow travellers are predominantly from occupational psychology and related disciplines there are two particular features of their work which both sets them apart from most of this group and which makes their work relevant to this thesis. The first is that they have focused specifically on supporting designers who are designing rather than computer system users in general. Their interests lie in enabling designers to carry out design tasks. The second distinction is that they build systems. They apply what they believe about design to themselves as designers of design support systems. Hence their research ideas, framed within a phenomenological outlook, are tested, demonstrated and challenged (effectively reflected upon in Schön's sense (cf. chapter 2 section 3.3)) by the building of what are termed "objects to think with". Janus, the system to support kitchen designers which is described below (section 3) is such an object.

1 Routine Design as a Generic Task : DSPL and AIR-CYL

At Ohio State University Chandrasekaran and his colleagues have been working for the last decade to build expert systems by identifying ways of organising and controlling knowledge that are generic, that is that can be used as high level building blocks, called generic tasks, for constructing and understanding knowledge based systems (Chandrasekaran, 1983, 1986). The description here is focused on one of these generic tasks called hierarchical design by plan selection and refinement, the high-level

language DSPL (Design Specialists and Plans Language) devised to represent it, and in particular on AIR-CYL an expert system for mechanical design of air cylinders implemented in DSPL (Brown, 1984, 1985, 1986a, 1989).

1.1 Background

The research at Ohio is based on the conviction that a key failing in expert systems is the lack of explicit representation of the problem at an information processing level. This failure is ascribed to the low level of abstraction supported by the languages and tools from which expert systems are constructed. "Most available languages, be they rule- , frame-, or logic-based, are more like the assembly languages of the field than programming languages with constructs essential for capturing the essence of the information processing phenomena" (Chandrasekaran, 1986, p.23).

Early work on diagnostic reasoning led to the idea that there are some ways of structuring knowledge and of controlling the way it is used which are common to diagnostic reasoning in different domains. On the other hand, it was expected that different kinds of problem solving activities (designing for example) would have different knowledge structures and control mechanisms, i.e different generic tasks associated with them. One major purpose of the research at Ohio is to identify generic tasks that can function as high-level building blocks for expert systems and to develop means (languages) for representing these generic tasks explicitly in expert systems in domains to which they apply. It is believed that given a new application, identification of the generic task(s) operative in the domain will lead to better constructed systems. The argument being that this approach will focus attention on design of the expert system at the right level of abstraction, give a framework for knowledge acquisition and enable richer explanations to be given by the system. The idea is that the high-level constructs from which such an expert system would be built would allow the problem solving being effected to be more clearly distinguished than it can be in the lower level constructs commonly used to build expert systems. The high-level building blocks would support representation of the domain knowledge and expression of the way it is used in a domain in terms appropriate to the problem solving task. So, for example, in diagnostic reasoning, examples of these terms would be ; malfunction hierarchies, rule-out strategies, and the setting up of differentials, whilst for artefact design examples would be ; component hierarchies and design plans. Essentially the idea is that control issues natural to the task should be plainly identifiable through the way they are represented in the expert system.

So far this research has resulted in the identification of a number of generic tasks. Examples of these are ; hierarchical classification (top down

refinement by movement through a hierarchically organised structure of concepts), hypothesis matching (establishing a concept by matching it to relevant data and determining the degree of "fit"), knowledge-directed information passing (inferring data values when not explicitly recorded from domain knowledge). MDX, an expert system for medical diagnosis (Chandrasekaran, 1983a) and the system associated with it for intelligent retrieval of data about a patient, PATREC (Mittal, 1984) can be viewed as a knowledge based system application which consists of the three generic tasks given as examples above.

What is of interest here, however, is the generic task associated with what Chandrasekaran calls "routine" design. It is this which forms the focus of the remaining discussion below. Before turning to this however, one other research theme needs to be introduced. This concerns the Ohio researchers' stance on the way knowledge should be organised in an expert system. Their belief is that knowledge should be distributed among communicating specialists, each specialist is defined to consist of domain knowledge (essentially factual material) and knowledge about how it is to be applied in problem solving (essentially procedural). The claim is that separating knowledge from the processes that use it gives a false sense of generality. They do not subscribe to the idea that a knowledge base can be shared in common by different problem solvers. This claim is backed up by analogies to how experts (specifically the medical community) is organized. The analogy briefly stated is that medical professionals are not separated into individuals who possess large volumes of "domain knowledge" and separate individuals who are expert at certain kinds of problem solving who somehow come together to tackle a particular problem by applying the problem solving skills of one to the domain knowledge "known" by another. Rather, the medical community can be seen as a collection of specialists ; clinicians, radiologists, pathologists, and so on, each of whom is able to "apply" domain knowledge to solve the problems which they are regarded as experts at solving (Chandrasekaran, 1983).

They argue that organising knowledge into specialists allows variety in representation structures for both factual and control knowledge to be permitted in a controlled way. By this means distinctions in control and inference between different parts of a task can be supported and made explicit in a way which is otherwise suppressed in systems which impose uniform control mechanisms (Brown, 1989). Use of specialists then, as originally conceived, results in expert systems being constructed on the basis of decomposition of the domain into sub-domains of expertise, each of which specialises in one kind of problem solving. In AIR-CYL, as will be described below, the idea of specialists is applied as an organising mechanism at a much lower level of problem decomposition than was originally conceived (and justified) by analogy with the organization of human professionals.

1.2 Research Purposes

The development of DSPL and its application to air cylinder design was motivated by a desire " to produce a generic theory of one type of design, and support it with an architecture and a high-level language that is suitable for that type of design". Application to the design of air cylinders was intended to embody a theory of one kind of design, called routine design, and to demonstrate its viability (Brown, 1989, p.31). The theory of design was intended to be domain independent so that a generic activity could be identified, reusable in other design applications of a similar kind. A generic task to add to the collection of generic building blocks, one which is seen to be common to routine design problems was identified and named "hierarchical design by plan selection and refinement".

Routine design is defined as a class of design where :

> the structure of the artefacts being designed is known, and can be hierarchically decomposed into parts (components);
>
> knowledge about how to design the components is available and can be described in terms of plans for designing them (these may be partial plans since components may be designed by designing sub-components - for which there will, in turn, be plans available);
>
> knowledge about how plans for designing (sub)components can fail and how to recover from failure is available.

Routine design characterised in this way can therefore, by definition, be carried out by a process of plan refinement. Routine design is seen to be the solving of problems where :

> design requirements and the goals to be achieved are fully specified at the outset;
>
> the components which are to be combined to produce the designed object and the functions to be carried out are known in advance and the interactions between them are at least weak if not negligible (the structure is essentially hierarchical, interactions can be handled by specific pre-established strategies (Brown, 1989, p.52));
>
> an effective way of decomposing the design problem in order to solve it is known in advance;
>
> the sub-problems (resulting from the decomposition) can have plans associated with them a priori (selection of an appropriate plan can be determined dynamically on the basis of how a design is progressing) so planning does not need to be carried out during design - just plan

selection;

failures can be predicted and strategies for coping with them can be pre-specified.

Two claims are made about routine design as defined above. The first is that routine planning and routine design are similar activities in the sense that both are attempting to produce a product to achieve some goal and both activities are concerned with construction or synthesis. (To illustrate this it has been demonstrated that an expert system to plan tactical missions for military aircraft can be constructed using DSPL.) The second claim about design defined this way is that "a significant proportion of everyday activity of practising designers falls into this class" (Chandrasekaran, 1989). DSPL has been developed to represent the generic task of problem solving defined as routine design. The intention of the researchers is to move from this routine design towards more complex design. "We believe that incremental exploration from routine design towards creative design is the best approach. This will allow discovery of most of the ingredients of a more complex form of design prior to investigating it" (Brown, 1989, p.130). This assertion is discussed later in section 1.9 of this chapter. The researchers' belief is that "difficult design problem solving" cannot yet be handled because an architecture of generic tasks to construct the complex task has not (yet) been found" (Chandrasekaran, 1986).

Routine design is taken be an activity consisting of four phases. These are ; requirements checking (making sure there is nothing missing or inconsistent in the specification), rough design (deciding or checking important attribute values, ones on which the rest of the design depends) , detailed design, and redesign (designing parts of the artefact again after failure to satisfy (a later) part of the design). A routine design task represented in DSPL is broken down into a hierarchy of sub-tasks, some design decisions being made at each level in the hierarchy. Movement through the hierarchy to lower levels represents refinement of the design. Design plans are associated with each specialist, these describe how the specialist can achieve its (sub)task. DSPL contains constructs for representing: specialists and plans, lower level primitives (tasks and steps), plan selection criteria, procedures for making selection of plans dynamically during design, and failure situations and how to recover from them. These are described more fully below in sections 1.4 and 1.8 in the context of the expert system AIR-CYL which applies DSPL to routine design (as defined above) of a mechanical device, an air cylinder.

1.3 Roles of User and System in Interaction

AIR-CYL (and any other similar routine design system constructed from DSPL) is intended to *do* the designing. The idea of design problem solving that is embodied in AIR-CYL is that of a cooperative activity

between specialists, specialists can be different types of problem solvers each of which solves a sub-problem using knowledge and inferences of specific types. The specialists are identified with clearly defined information processing responsibilities. The belief is that the function-determined architecture of this approach makes clear the separation of the contributions of the system and the human user, "...whenever knowledge and control can be explicitly stated for one of the modules or building blocks, that module can be built directly, by using a knowledge and control representation that is appropriate to that task ... if knowledge for a module is not explicitly available, the human can be part of the loop for providing information that the module would have been responsible for" (Brown, 1989, p.31). Thus, in AIR-CYL, when a sub-task (specialist) needs information that cannot be supplied by another sub-task a human can be asked to supply it. The idea is that once the task is better understood more of what the human is supplying will be provided by the system - it will undertake more sub-tasks without human assistance. The form that the user-system interaction takes is therefore initiation of requests for information from the user when there is no other way for the system to obtain the information it needs.

These roles for system and user strongly contrast with the roles defined for interaction in the other two systems described in this chapter. Indeed, Fischer (one of the principle authors of the system described in section 3 of this chapter) intends criticism of expert systems' interaction with users when he describes the classical expert system approach as one of asking the user for input and then returning an answer (Fischer, 1990a) - however, this description aptly summarises the roles of user and system in the approach taken with AIR-CYL.

1.4 Main Architectural Components

AIR-CYL, like any other expert system for routine design based on the same architecture, is organised as a hierarchy of cooperating specialists. Each specialist is responsible for a particular part of the design. Specialists towards the top of the hierarchy are concerned with general aspects of the artefact being designed, whilst the lower level specialists handle design of more specific components or portions of the air cylinder.

Specialists control the problem solving associated with their part of the design. In AIR-CYL the specialists are defined in a hierarchical arrangement which is intended to reflect the conceptual structure of the design problem as a human designer sees it. (Details of how this was elicited are brief - the designer was interviewed and protocols were analyzed (Brown, 1989 p.108).) A specialist has a set of plans associated with it and a plan selector. Plans are selected depending on the current state of a design in progress, so specialists can be thought of as refiners of plans in which they themselves appear. Specialists also have rough plans associated with them. These are the plans used for the rough design phase

which may precede the detailed design phase (stated in terms of the phases of routine design as defined in section 1.3 above).

A plan is a sequence of calls to specialists (or to tasks - which are described below) and may include tests which represent constraints which need to be satisfied at particular stages (of refinement) in the design e.g. a constraint may represent knowledge about what must be satisfied by a specialist before it can be considered to have successfully completed its contribution to the design as a whole. A plan is one way of designing the part of the air cylinder for which the specialist to which it is attached is responsible. Where there is more than one way for the specialist to carry out its responsibilities, more than one plan will be associated with it. Each design plan has a design plan sponsor associated with it. This contains knowledge about how to determine the appropriateness of a plan on the basis of the state of a current design (i.e. in the run-time context). A specialist's plan selector contains knowledge about how to select a plan on the basis of the judgements of the design plan sponsors. The core of the control knowledge represented in a specialist is thus of three types represented by design plans, design plan sponsors and design plan selectors.

Procedures which make a single design decision are termed design steps. A step will typically establish a value for one of the attributes of the air cylinder or of one of its component parts. Steps are grouped into larger functional units called tasks. Tasks in this sense are just organisational units which perform a group of steps (and perhaps test constraints) associated with something logically or structurally coherent in terms of the design. Knowledge is distributed throughout the system about what to do when failure occurs during a step, task, constraint test, or when a specialist fails to produce its contribution to the design. This knowledge is located at the places where it is to be applied, i.e. where it can have an effect. The handling of failure is described briefly in the next section on system dynamics and is discussed in detail in section 1.7.

As design progresses values of the attributes of the components of the air cylinder are decided. The design database records these values in a layered manner, the layers represent the level of commitment with which each attribute value has been established. A design step makes a single "alteration" to the design when it determines a single attribute's value. Once the values of a coherent collection of attributes has been established, usually through the successful completion of a design task (a task may undo some alterations along the way in order to be able to successfully complete), the resulting collection of alterations is classed as a "revision" to the design. Revisions themselves may need to be undone if a later part of a plan fails and causes redesign. Once a plan has successfully completed, however, the collection of revisions associated with it will be used to make a new record of the design. This is termed a "drawing". The design database is structured so that alterations, revisions and drawings are distinguished.

Values sought by, for example, steps, constraint tests, or tasks, are first sought among the alterations and revisions outstanding as updates to a drawing. This layered database structure is intended to allow changes to the design, prompted by failure in the design process, to be made in an efficient manner.

1.5 System Dynamics

Design commences at the highest level specialist in the specialist hierarchy. Beginning at this specialist, each specialist selects a design plan appropriate to the specific design problem's requirements and from consideration of the current state of the solution. Selected plans are executed by performing the actions (tasks, steps, constraint tests, invocation of (sub) specialists) which are specified in it. When a design plan refers to a lower level specialist, the plan is refined by passing control to that (sub)specialist which in turn selects a plan to carry out its responsibilities with respect to the overall design. When a specialist is entered i.e. is given control, each of its plan sponsors is executed by the DSPL interpreter. Plan sponsors determine the appropriateness of the plans to which they relate by examining the design database (the current design) and other relevant information - for example information about the design requirements. Once this has been done, the plan selector associated with the specialist makes a choice of which plan to execute on the basis of the suitabilities of each plan as recorded by each plan's sponsor. The plan selector returns the name of the chosen plan to the specialist which then executes the plan.

During design, failure at any point is handled in a "bottom up" fashion. That is, failure is handled as locally as possible on the basis that it is desirable to make as few changes as possible to the current design to overcome the problem. When failure occurs, information about the failure e.g. the amount by which an attribute value exceeds a set tolerance, is incorporated in failure suggestions. Failure suggestions are attached to constraints, steps and tasks for use locally within them. When a failure cannot be handled local to where it has occurred failure suggestions may be passed up to a higher level for use where they can be applied. Knowledge about how to handle failures is placed in specialists, tasks, and steps. In each case this knowledge handles messages from failing subordinates, or from itself. Knowledge of this sort is termed failure handler knowledge and takes the form of "situation-action" associations for recognising failures and deciding what action to take.

Two kinds of redesign arising from failure are distinguished. One is the redesigning associated directly with a failure to design a part of the air cylinder. Plan sponsors use information about what plans have been tried already during design to determine whether their plans should be recommended. The second kind of redesign is that found necessary when

part of a design has to be redone because of a failure elsewhere in the design. Recovery from failure (at a high level) is achieved by plan selection (i.e. selecting a different plan) and by redesigning some piece of the air cylinder to alleviate the source of failure. The authors of DSPL believe that strategies for recovery tend to be problem independent at this level, i.e. applicable to design problem solving fairly generally. At a low level (say a step), redesign consists of altering the value of a parameter and here is believed to be problem dependent. What happens is very much determined by the particular (current) design problem the system is solving.

In the air cylinder design application fifteen distinct parts of an air cylinder are designed for each new product. At the outset of design nineteen values are provided as input to the design process. Organization of the air cylinder designer system into specialists largely consists of a decomposition of the task into the design of the main assemblies of the air cylinder (the head, the spring, and the rest), these are further decomposed into their composite parts where appropriate. Brown makes the observation that this decomposition was arrived at through interviewing an air cylinder designer and through examining protocols, "on examination we could see that this organization tends to localise dependencies, and allows for parallel design activities" (Brown, 1989, p.109). The task structure defined for the design of air cylinders reflects the four phases of routine design (defined in section 1.2 above). The detailed design phase is decomposed into specialists reflecting the conceptual structure associated with the decomposition of the air cylinder into assemblies and components as described above.

1.6 Handling Design Trade-offs / Relaxing Constraints

Design trade-offs are handled in a restricted sense by AIR-CYL. Strictly speaking trade-off does not occur; it is rather the case that redesign is carried out under the initiation of the failure handlers when it is found that a design requirement cannot be met (later) in design without redesign of another part. What actually happens is that a different plan is selected following a failure, perhaps one that will lead to design of the part causing difficulty earlier than was initially scheduled. AIR-CYL, in carrying out routine design as it has been defined above, operates by selecting and refining predefined plans on a basis determined by the current design problem. The task structure defined using the constructs of DSPL and the way the DSPL interpreter operates require that the design problem is such that it is possible to establish "the exact order that the designer uses so as to be sure that all the required values are available before each agent acts" (Brown, 1989, p.127).

For any design problem where there are interdependencies between specialists to any real degree, a DSPL decomposition of the design task into specialists with separate realms of responsibility within the overall design would be problematical. The authors make a gesture towards

acknowledging this, although their comment is given in the context of considering performance degradation. "If there are many mutual dependencies then the system will be making a lot of early decisions that may be based on limited information and may well be wrong, leading to poor results from the design phase. If the dependencies do not allow the design knowledge to fall into easily divisible groups then it will be very difficult to produce a design system at all." (Brown, 1989, p.128) Decomposition of design into specialists is discussed further in section 1.9 of this chapter.

Relaxation of constraints (requirements) is not directly supported in AIR-CYL (or representable in DSPL). The definition of routine design demands design requirements to be fully specified at the outset (before "design" begins). In AIR-CYL the design requirements are represented as the initial input values presented to the system, presumably therefore, in the event of a (total) failure to design an air cylinder given these parameters, the user is free to (re)try with another set of input values. However, it is possible to relax criteria for selecting plans by specifying preferences in plan selectors. Sponsors evaluate the appropriateness of their plans and label them qualitatively from a spectrum of "perfect", "suitable", ..."unsuitable", if no plan sponsor produces a "perfect" plan that the selector can choose, the selector can be directed to select a plan marked "suitable" by its sponsor.

1.7 Generation and Evaluation of Alternatives

Routine design, as defined in the DSPL context, requires *completely* specified requirements at the outset, clear goals, and a set of plans established *a priori* which can be used to achieve the goals. The idea of producing a number of alternative designs or of pursuing a number of routes through the design plans is not entertained in a DSPL designer. When a plan fails a form of backtracking can occur followed by redesign using another strategy for exercising a specialist's domain and problem solving knowledge.

The complete task structure produced for the air cylinder design application includes a high level (final) task of evaluation of the design. This is decomposed into separate sub-tasks representing evaluation of the weight and cost aspects of the air cylinder design. Evaluation of this sort was observed to be an aspect of the design process and appeared sometimes to result in modifications to the design (when carried out by human designers). The evaluation aspect of the design task was ignored however in the implementation of AIR-CYL. "Air cylinder design concluded with some cost and weight evaluations and some subsequent modifications to the design. As these modifications were not making substantial changes to the design they were ignored for the AIR-CYL implementation" (Brown, 1989,

p.125). The authors of AIR-CYL are not themselves clear about whether if post-design evaluation turns up unsatisfactory aspects of the design the redesign processes of DSPL can or cannot be used to handle this aspect of the design process (Brown, 1989, p.126).

There is no indication given of a need to compare alternative designs and evaluate these against one another. All proposals for evaluation imply quantitative or boolean evaluation of attributes of the design against absolute (external) values representing for example cost, weight, or manufacturability.

1.8 Support for Justification of Design Decisions

DSPL is intended to provide high level constructs for representing domain knowledge and control strategies in terms appropriate for routine design task. The intention is that control issues natural to this (generic) task should be plainly encoded. In DSPL different kinds of knowledge are distinguished directly by their association with the language constructs. For example, plans, plan sponsors and plan selectors are distinguished in DSPL as is knowledge about how to handle failures both those arising locally and those being communicated (upwards) from lower levels through association with failure suggestions and failure handlers in steps, tasks, and specialists. The design task (overall) is explicitly represented as a hierarchy of specialist each of which has responsibilities for control and application of the knowledge associated with components of the task structure.

By these means AIR-CYL and other systems constructed from DSPL have the potential to provide explanations of the kind supported by any second generation expert system in which task structure and strategies are explicitly represented. "In expert system problem solving there exists a close relationship between structures of understanding and explanation capabilities. For a system to 'understand itself', it should be able to examine its own knowledge structures, its problem solving strategy, and its problem solving behaviour particular to a specific case" (Chandrasekaran, 1989, p.391). The explanations given by a DSPL system are based on the idea that the agent (specialist, design plan, plan selector, plan sponsor, task, step or constraint) which makes a decision is responsible for explaining it. Each agent answers questions about what it has done. It also handles "why-type" questions by referring to agents from which it has received control and "how-type" questions by referring to agents to which it has passed control . Explanations of these kinds in DSPL are instantiated by making reference to the run-time trace which records the system execution relating to the current design (Chandrasekaran, 1989). ("Why-type" and "how-type" explanations of second generation expert systems are discussed in chapter 4 section 2.3 and their relevance in supporting design in particular in chapter 5 section 3.3.)

AIR-CYL has rich potential for giving explanations of its design decisions but it has no means of justifying these. In later work, on the application of DSPL to military aircraft tactical mission planning, proposals have been made to support justification through use of structures to explicitly represent what is termed "deeper understanding". In the case of mission planning this is construed as understanding of how the plans work (Chandrasekaran 1989). Essentially, a knowledge structure for devices in which structure, function, behaviour and assumptions are represented (Sembugamoorthy, 1986) is proposed to provide this "deeper" understanding. (A device is defined as being any structure which serves a purpose. Hence a plan can be viewed as an abstract device.) The problem with looking to "deep" models to provide explanation and justifications for decisions is discussed in chapter 4 section 4.1[1]. The limitations expressed there are followed through in terms of the demands which can realistically be made of knowledge based system technology in supporting design are considered in detail in chapter 5.

1.9 Discussion

Two aspects of DSPL/AIR-CYL are discussed further in this section. The first concerns the validity of decomposing a routine design task into specialists The second concerns the wider issue of the idea that it is possible to learn about building systems to do complex design by studying how to build systems to do routine design.

1.9.1 Design decomposition

The rationale for decomposing tasks into cooperating specialists where each specialist consists of domain (factual) knowledge and knowledge about how it is to be applied in problem solving is based on the claim of analogy to how a community of professional (the medical community) organises itself. In the case of designing air cylinders, however, the idea of specialists has been taken to a much finer level of granularity than that supported by the original analogy with human "problem solvers". In the case of air cylinder design by humans *two* designers cooperate. One of these designs the spring, the other the rest of the air cylinder. In AIR-CYL the top level decomposition presented of the main design phase not only gives specialists for designing the spring, the head of the cylinder and "the rest of the cylinder" but considerably more specialists through further decomposition. For instance, "the rest of the cylinder" is further decomposed into four

---

[1]Here the problem of trying to explain and justify on the basis of distinguishing structure, function, and behaviour can economically if cryptically summarised by quoting George Steiner on the dilemma of the hermeneutic circle "we attempt to define a thing by the use of attributes which already presume a definition" (Steiner, 1978, p.26).

components (cap, piston and rod, tube, and bumper), i.e. further sub-tasks each carried out by further specialists.

The decomposition used in AIR-CYL is justified on the basis that it was elicited from a designer and can be seen to localise dependencies thus allowing design activity to proceed in parallel. In AIR-CYL, as implemented, design decisions are organized so that all the information needed for a sub-task is available when the sub-task is reached and also constraint tests are minimised and placed as locally as possible. The ability to do these two things enables AIR-CYL to do its designing. However, there is evidence which must lead to the questioning of the validity of the decomposition of the design in this way. Firstly, although it was established beyond doubt (by the researchers) that nineteen values are used as input to the design they are not all used in AIR-CYL "due to the limited amount of debriefing of the designer and our ignorance about air cylinders" (Brown, 1989, p.110). Secondly, as has already been noted above, some of the design activity observed is not done by AIR-CYL, namely, the task of evaluating the design against the global constraints of weight and cost. The researchers themselves note that the designer, at times, exhibits design behaviour which does not fit the decomposition of the task represented in AIR-CYL. Sometimes the designer was observed to "tweak the design slightly, by propagating small changes in dimensions backwards and forwards along the longitudinal axis of the air cylinder until the desired state was reached. ...These changes appear not to be based on the same conceptual organization of the problem" (Brown, 1989, p.132).

These two observations, particularly the second one, seem to suggest that the scope for, the merit of, and even the validity of, decomposing even the very limited kind of design defined as here routine design is questionable. The authors of AIR-CYL identify decomposition as a research topic in terms of a need to study "how designers decompose design problems in order to handle complexity" and a consequential research problem of understanding "how to compose the solutions to the sub-problems produced by decomposition" (Brown, 1989, p.131).

Read differently the observations of the air cylinder designers' behaviour and the interviews with them might be interpreted as further support for the conclusions from research into what designers actually do presented in chapter 2 .

1.9.2 Moving on from routine design

The idea that it is possible to learn about more complex design tasks by moving on from studying routine design seems questionable at several levels. Firstly, there is the matter of what is outside the scope of routine design. For example, the definition of routine design sets aside any ideas that design is an ill-structured and wicked problem (chapter 2 section 2). It addresses problem solving defined as starting with a completely well

specified set of requirements, having a clearly defined goal and a pre-defined set of options to choose from (notwithstanding that choice depends on problem-specific data) at each stage of design. These are not characteristics of more complex design. It is difficult, therefore, to see how light will be shed on applying knowledge based systems technology to complex design by building systems to do routine design defined in this way.

Secondly, there is the way that routine design of air cylinders has been handled. The air cylinder design problem has been carefully selected (or at least the scope of AIR-CYL has been carefully restricted) to largely exclude interactions between specialists. (In fact, there appear still to be some, despite these efforts, but these are handled by making sure that it is possible to predict exactly when, during the design process, specific constraint tests are to be applied.) The design problem decomposition into specialists for AIR-CYL, and the constructs of the DSPL language which have developed do not therefore explore how tasks where interactions are an inherent aspect can be represented or even what the requirements of such a representation might be. (In chapter 4 section 3.1 this shortcoming proves to be a serious one for modelling designer's expertise.) Thirdly, there is the related issue of investigating relaxation of design constraints or design requirements. Once again it follows from the definition of routine design that since design requirements are available *completely specified* at the outset - design requirements are values input to the AIR-CYL system as initial "givens" - that reasoning about them in any interesting sense is therefore beyond AIR-CYL's scope. Finally, the role of evaluation in design, one which appears to be critical (to what constitutes the design task overall), has been ignored in AIR-CYL. The design process embodied in DSPL's constructs and interpreter excludes components which can be used to explore the role of evaluation in design. It is therefore difficult to see how systems which carry out routine design defined in such restricted terms can lead to discovery about more complex design problems in an incremental fashion as claimed by researchers following this approach (section 1.2 above).

2 Supporting Exploratory Design : EDS

Since 1984 a programme of research in the field of A.I. in design has been underway in the Department of Artificial Intelligence at the University of Edinburgh. This body of work and in particular the architecture for a knowledge based system to support engineering design has come to be known as the Edinburgh Designer System (EDS) (Smithers, 1986, 1989, 1990, 1992). It is this design support system which is described here.

2.1 Background

2.1.1 Research environment

EDS was initially conceived as the central component of a large scale demonstrator project within the Alvey research programme initiated in the early 1980s (Alvey 1982). The work on this demonstrator which was called "Design to Product" (Smithers, 1986) aimed to show how computer aided design (CAD), computer-aided manufacture (CAM) and flexible manufacturing systems (FMS) could be integrated with one another. EDS was intended to maintain and manage knowledge about electro-mechanical products from the initial, schematic design phase through detailed design, manufacturing planning, manufacture, testing, up to and including servicing, maintenance and product de-commissioning. EDS was intended to integrate all stages from design to production. The idea was to develop a uniform means of managing and manipulating the knowledge generated during the life-cycle of a product to show that a more efficient, flexible manufacturing environment could be achieved through elimination of the knowledge losses associated with the usual practice of (just) linking a variety of essentially separate computer based systems (e.g. CAD, CAM, robotic assembly cells). The "Design to Product" project aimed to demonstrate the feasibility and advantages of a flexible manufacturing system that was based on an integrated repository of knowledge about the product - which EDS would provide.

EDS has evolved over the years since it was first conceived as the core to the original research programme. There have been six distinct implementations of the system in Poplog and applications have been extended from the mechanical engineering domain to the design of chemical compounds - new pharmaceutical drugs (Smithers, 1992) and proposals have been made to apply the central ideas from EDS to support the design of VLSI-based electronic devices and the design of software.

2.1.2 Theoretical stance

Current work on EDS is based on the belief that designing is essentially a process of exploration which involves redefining the problem, that is of structuring it as well as solving it (i.e the problem setting and problem solving as described in chapter 2 section 3). Design is seen as solution-oriented in the sense that designers are believed to explore solutions to find out more about the problem (cf. chapter 2, particularly the work of Lawson and Schön). Design requirements are *assumed* to be initially incomplete and possibly inconsistent. The exploration of the incompleteness and the inconsistencies are considered to be part of the design process. Exploration is taken to be non-monotonic, and additionally it is assumed that a number of alternative lines of reasoning for possible progress may be operative

simultaneously.

In the initial stages of the development of EDS its architecture was not founded on an explicit model of the design process, rather the architecture was determined pragmatically as a means of integrating a number of functionally distinct (sub)systems which were ready to hand from previous research projects. However, the architecture which has evolved is not incompatible with the view of design as exploration nor is the mode of interaction which is supported by it. Earlier research projects which contributed to "Design to Product" as a whole were an equation solving system called PRESS (Bundy, 1979), a robot programming language, RAPT (Popplestone, 1980), a system for representing the functional behaviour of components and verifying this by comparison with a specification called VERIFY (Barrow, 1985), and some others including a system for solid modelling of moving parts.

More recently one of the driving forces for further development of EDS has been a desire to develop a computational theory of design centred upon the idea that the essential nature of the design process is exploratory (Smithers, 1990). The authors of EDS do not subscribe to the view widely held among the A.I. research community that design is (just) search in a very large problem space. They give two main reasons for opposing this. The first is that since the goal (the design requirements) are initially both incomplete and often also inconsistent the idea of goal-directed search cannot usefully be applied. Secondly, on the same basis, the start state is also not clear in design tasks. Effectively then, in design, part of the problem, and therefore part of the design process itself, is concerned with deciding what problem is to be "solved". In their argument against representing the design process as search of a problem space we can see echoes of Jones' observations about the limitations of design methods (chapter 2 section 1). EDS tackles the idea that design is a wicked problem (chapter 2 section 2) head on, and thus it contrasts radically with the work exemplified by AIR-CYL described in section 1 of this chapter and it is through this perspective that it makes its strongest contribution to research in this field.

Whilst the authors of EDS deny that designing is equivalent to computational search, they do claim that some other formalism, which they term computational exploration, should be sought to account for the design process. They are firmly based in the A.I. community to the extent that they espouse the belief that "A.I. is a science concerned with understanding intelligent behaviour and how it can be *created* artificially" (Smithers, op.cit., p.11, emphasis added). Their search for a formalism includes an explicit claim far stronger than a modelling claim, since they seek to build and test A.I. programs "to realize computational processes which are discovered to engender the kinds of intelligent behaviour" (Smithers, op.cit.) in which they are interested. (Modelling claims are dealt with chapter 4 section 1.4.)

2.2 Research Purposes

The motivation for building successive versions of EDS is described as an attempt to understand design by building A.I.-based design support systems for human designers. The rationale given for building design support systems rather than systems which actually do the designing artificially has two aspects. The first is that, since design as a kind of intelligent behaviour is poorly understood, the demands of a system to support design are fewer or at least more controllable, and that design support reduces the demands for detailed understanding of design on the grounds that not all the reasoning has to be done artificially. This point is discussed in section 2.9 of this chapter. The second aspect is that by aiming to *support* design it is possible to study design tasks of a larger scale and of greater complexity than can currently be usefully attempted by fully automated systems. Constructing support systems is thought to give scope for exploring and testing what is already understood about design. Design is acknowledged to be complex and the A.I. route is felt to be just one among many possible approaches. The programme of research within which EDS is being developed is intended to contribute to an understanding of design as a knowledge-based kind of behaviour. Effort is focused on exploring "to what extent design involves explicit intelligent knowledge-based behaviour and in how much this can be supported by or carried out by artificial systems" (Smithers, 1990).

In the evolution of EDS a subtle shift in the purposes of the research can be seen. At the start, EDS was developed as the heart of the "Design to Product" research programme. Reports from that era show that it was believed that the problems (to be overcome) were understood, the technology was available, and what was needed was the construction of a system that could demonstrate how a useful application could be developed - a pre-commercial demonstration of what could be done and the benefits to be had from doing it. More recently the purposes of the research which development of EDS is reported to support is framed in terms of three different levels of enquiry into knowledge based design based on Newell's analysis of knowledge and representation (Newell, 1982). The three levels are; knowledge, symbol and system[2]. The stated research purposes governing EDS development are specified at each of these three levels as follows. The model of design (the exploration model) is "intended to serve as a mechanism for developing and expressing our understanding of how knowledge is organized (explicitly and implicitly), and how it is used and

---

2 According to Newell, at the knowledge level an account is given in terms of what knowledge is used to perform tasks which require intelligence. At the symbol level an account is given in terms of syntactic operations which result in intelligent behaviour at the system level, whereas the system level is concerned with how a system can be built with current technology to perform the syntactic operations.

generated in design" (Smithers, 1990 p.80). The architecture of EDS is to express an understanding of design at the symbol level since the architecture of the system defines syntactically the knowledge represented and used to actively support design. The development of the architecture is intended to demonstrate (and test) how much of the design process can be characterized as symbol processing and how much can therefore be supported or automated by computer systems. At the system level the realization of components of the EDS architecture as software products is to demonstrate what can and cannot be built, how this can be done, and also to contribute to an understanding of the nature of any limitations which emerge.

2.3 Roles of the Designer (User) and System in Interaction

EDS is a system which *supports* a designer. A designer is broadly defined since EDS, as originally conceived, supports decision making about all the stages of a product's life from design onwards. The underlying belief is that the design process is exploratory and thus involves defining, redefining and refining the design requirements as well as constructing a satisfactory design solution. EDS accepts assumptions declared to it by the designer. These declarations are about both design requirements and solution (design) possibilities. EDS assists the designer by inferring the implications of the designer's assumptions and communicating these consequences to the designer. The designer then decides what to do next and communicates this to the system in the form of further declared assumptions which represent design decisions. Different kinds of inferences are made by different knowledge sources which form components of the EDS architecture. (The main architectural components of EDS are described in the next section.)

The user is viewed as a knowledge source whose contributions always take the highest priority. The user can direct the focus of EDS by specifying which knowledge sources are to participate in making inferences from assumptions. The user is relied upon to know when computationally expensive inferences from a particular knowledge source are relevant to his current exploration. Inferences which are low in computational cost are generally invoked automatically (Logan, 1992)

In exploratory design a number of distinct alternatives may be pursued by a designer. EDS supports this by permitting a number of distinct contexts or views to be maintained for the same design in progress (Logan, op.cit.). Contexts contain information relating to part of a particular design solution or task. Using the context manager the user can partition information to reflect his interests or goals and thus he can be supported in pursuing a number of alternative ideas or lines of reasoning. By selecting a particular view or views the user can focus the system's attention, and can restrict EDS inferencing (knowledge source operation) to particular sets of information which belong to the current view the designer is exploring.

Thus, by changing contexts, the user can focus the system's attention on a particular part of the problem or on a particular kind of inference.

Once a design is finalised the description of it must be consistent if it is to make sense. However, EDS can accommodate the fact that *during* the design process the description of a design will contain inconsistencies while the designer explores possibilities that will satisfy the design requirements adequately. The designer is therefore not constrained to proceed monotonically towards a final design along a single line of reasoning for this would not allow exploration of partial solutions and alternative design possibilities. EDS uses an assumption-based truth maintenance system (de Kleer, 1986) to trace the set of assumptions on which a particular piece of information depends. During design inconsistent decisions are permitted to exist simultaneously, the truth maintenance system is used to maintain environments (sets of assumptions) with which inferred pieces of information are labelled. Ultimately the procedures of the truth maintenance system restore consistency by producing sets of internally consistent design decisions which are presented to the user.

The role of EDS in supporting the designer during the design process is multi-faceted, the main aspects of the support can be summarised as :

acting as a repository of domain knowledge and presenting this to the user at opportune moments during design;

inferring the consequences of the designer's assertions (assumptions) and using default reasoning to relieve him of some routine reasoning;

maintaining a record of the designer's assertions, the inferences made, and the dependencies between them;

maintaining consistency between inferences, alerting the user to inconsistencies;

helping to manage the complexity of the task by recording and structuring the information generated during design;

supporting and managing contexts for the user so that multiple avenues to a design solution can be explored at one time, providing the user with means of examining mutually inconsistent solutions.

2.4 Main Architectural Components

The main components of EDS are the domain knowledge base, the design description document, the consistency maintenance system, and two distinct classes of inference engines known as generic ones and special

purpose (domain specific) ones. The overall functionality of EDS can be understood from descriptions of these main components although there are other components which handle interaction amongst the main components and between EDS and its user and there are a number of other interfaces (e.g. syntax checkers and debugging tools) which are not described here. Each of the main components is described below and their relationship to one another is depicted in figure 3.2. Further information on how the components relate to one another dynamically is given in the next section (section 2.5).

The *domain knowledge base* is composed of two main closely related components, the domain knowledge and knowledge about how to use this in design - the design knowledge. The organization of the domain knowledge takes the form of a taxonomy of structured objects. Each object represents a functional unit which can be selected on the basis of its behavioural properties for inclusion in the artefact being designed. (This structure is a legacy of the VERIFY system mentioned in section 2.1.2 above.) The functional units, called module classes, are related to one another by representations of both part-of and kind-of relationships. Each module is represented as a structure which has properties and processes associated with it. Design knowledge is dealt with rather cursorily in the literature on EDS. It is described as being derived from previous designs and as being concerned with how the domain knowledge is used to explore design possibilities, to define and solve design problems (Logan, 1992b) and as comprising relevant design methods and strategies (Smithers, 1990). Much of the design knowledge appears to be implicit in the structuring and the representation (i.e. the choice of properties and processes) attached to the domain knowledge and in its taxonomic structure. However some of the design knowledge is elsewhere, explicitly and implicitly represented in both the knowledge sources and in the ATMS procedures. The representation of design knowledge in EDS is discussed further in section 2.9 of this chapter.

A *design description document* is created for each specific design task. It consists of three components. These are representations of the exploration of the design space (the design history), a representation (eventually) of the final design, termed the design specification, and a representation of the design requirements which have evolved during the exploration and which are satisfied by the final design specification. These three components are represented as dependency networks, the nodes of which are generated and truth maintained by the assumption-based truth maintenance system (ATMS). The operation of the ATMS is described below in the next section.

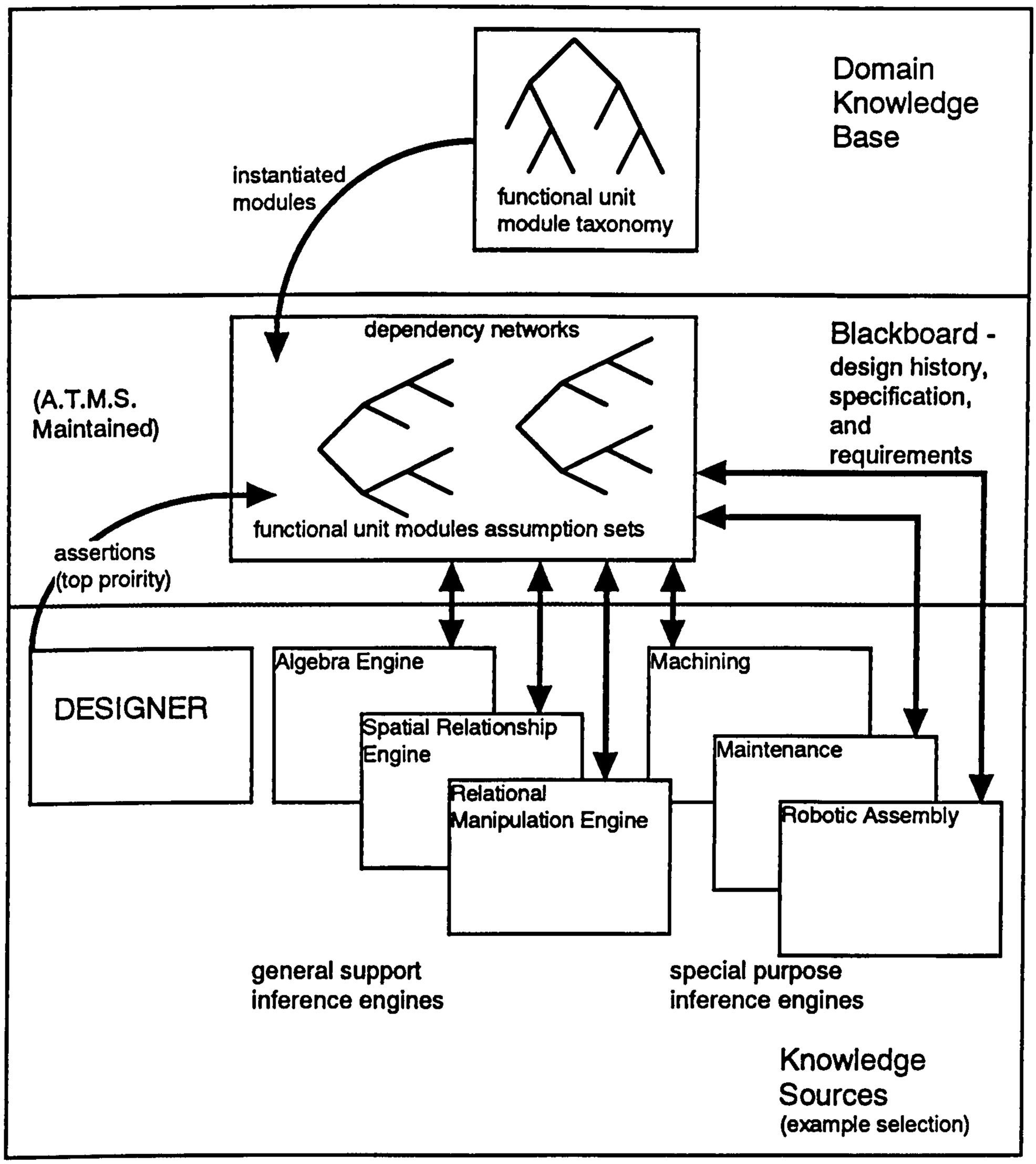


Figure 3.2 Main components of EDS in "Design to Product" research programme.

The design description document is the information centre of the system. Its contents are read and modified by the inference engines as design proceeds. EDS operates as a variant of the blackboard architecture (Engelmore, 1988) in which the design description document forms the major part of the blackboard and the inference engines operate as knowledge sources. The *consistency maintenance system* uses the ATMS to truth maintain the contents of the blackboard and also supports the alternative assumption sets, the environments which allow the user to explore the design through different contexts as described in section 2.3

above.

The *inference engines* fall into two categories. All the inference engines are able to make inferences on the basis of the contents of the design description document and to effect additions to it. The first category, called general support inference engines,is largely inherited from work which pre-dates EDS (see section 2.2 above). The general support inference engines most consistently mentioned in the publications on EDS are ; the relational manipulation engine which performs relational algebraic operations and interpolation of values given data in tabular form, the algebraic manipulation engine which solves sets of linear equations , the spatial relationship engine which is capable of spatial inferencing, and an evaluation engine which can simplify equations and propagate values (e.g. for constraint satisfaction). In early reports a further general support inference engine capable of inferring data about solid geometry is mentioned. However limitations in current techniques for solid modelling seem to limit the ability to reason about shapes (partially defined ones at least).

The second category of inference engine, termed special purpose inference engines, is described in early reports (Smithers, 1986) as sources consisting of specialised knowledge for different kinds of activity e.g. knowledge associated with design for robotic assembly, design for machining, design for maintenance. However, it was acknowledged that some of " these specialised aspects of design will be supported partly by including relevant knowledge in the functional unit module taxonomy" i.e. in the domain knowledge base. No special purpose inference engines are included in EDS currently and there appear to be problems in integrating these into the EDS architecture because of restrictions on maintaining consistency (and detecting inconsistencies) within design alternatives using the ATMS as currently designed (Logan 1992 discusses this point in more detail).

In recent work on the application of part of EDS to the design of drugs (Smithers, 1992), a further variation of special purpose knowledge sources, called support systems, are proposed, each of which is to be associated with part of the design task.

2.5 System Dynamics

Designing with EDS typically commences with the user identifying something relevant from the initial (functional) design requirements and making an assertion about it to EDS. If there is something relevant (from a function perspective) in the functional unit module taxonomy of the domain knowledge base then a relevant fragment of the taxonomy will be instantiated and passed to the consistency maintenance system which will form it into nodes of the ATMS dependency network. By this means it becomes part of the design description document and appears on the

blackboard. Design proceeds through the creation of instances of module classes and through the assignment of values to the parameters of the instantiations. The designer may make assertions, e.g. assign values to parameters, the general purpose knowledge sources can then be invoked (asked) if they can infer anything from the assertions made by the designer or from anything else appearing in the design description document.

An interaction management system receives notification from knowledge sources of what they can do and what information from the design description document they need to be able to do it. The interaction manager controls an agenda of these notifications which are "bids" to infer. When a knowledge source is used to make an inference the outcome is recorded on the design description document in the form of additions to the dependency network and may in turn result in new bids for inferencing from further knowledge sources. Bids for action, known as knowledge source activation records (KSARs), are executed by the bidding knowledge sources until no new inferences can be made. KSARs are ranked on the agenda dynamically according to the current state of the solution (the contents of the dependency network of the design description document). Priority is attached to KSARs which contribute to the developing solution by giving priority to the most recent additions to the blackboard. The detection of inconsistencies is also given particular attention as these areas are likely to reflect inconsistencies in the design requirements or to indicate potential problems with an evolving design solution. The detection of conflicts and their consequential effects is treated in more detail in section 2.8 below.

The designer examines the outcome of the inferencing carried out by the knowledge sources by displaying parts of the design description document and asking for explanations of the inferences which have been made. (The designer can control the focusing of attention as outlined in section 2.3 above.) On the basis of this the designer decides what to do next. For example, he may declare further assumptions e.g. by setting the values of parameters or by invoking particular knowledge sources to pursue a line of reasoning further under user control (see section 2.3). The designer,the domain knowledge base and the knowledge sources which support him co-operate through interaction with the design description document which mediates all progress on the blackboard. The KSARs are truth maintained under the control of the ATMS within the consistency maintenance system. This means that false bids, i.e. ones that become inappropriate as a result of other inferences (KSARs with inconsistent antecedents) can be discarded.

A further dimension of complexity is added to the ATMS blackboard dynamics described above to cater for the simultaneous support of multiple inconsistent contexts while design proceeds. This has resulted in limitations to the operations that the ATMS can perform. The maintenance of different, but simultaneously held, lines of reasoning is described in more detail in section 2.7 of this chapter.

2.6 Handling Design Trade-offs / Relaxing Constraints

The making of design trade-offs and the relaxation of constraints is the responsibility of the designer. EDS supports these kinds of decisions in two ways. Firstly, the designer is notified of inconsistencies in design requirements and / or the emerging design solutions, so the system can be said to be able to draw situations requiring trade-offs to the designer's notice. The designer is also presented with consistent alternative sets of decisions (assumptions) to inform his choice of how best to proceed. Secondly, once the designer has decided what to try (and tries it by making an assertion e.g. perhaps specifying a change to a parameter to represent a relaxation of some constraint) the EDS domain knowledge base and the knowledge sources can be brought into action to infer the consequences of the designer's decision and to communicate these to him. When a user needs to relax a constraint (by changing a requirement, changing the value of a parameter, or taking a different approach) he is assisted by EDS which provides utilities for examining the contents of the design description document. For example, a designer can view, on a graphical display, a representation of how a parameter was inferred.

2.7 Generation and Evaluation of Alternatives

Support for the generation of alternative lines of reasoning to creatively explore the design possibilities is a central feature of EDS. It is one of the areas where EDS makes its strongest contribution to research in knowledge-based design support systems. Multiple inconsistent contexts can be supported simultaneously as design proceeds. The user can control both his and the system's focus of attention to a particular view or views at any point during the design process. A view is termed a belief set since the assumptions (representing design requirements, domain knowledge and decisions made by the designer) that define it do not have to be mutually consistent. The views mechanism allows the design description (document) to be partitioned so that alternatives and their implications can be explored by the designer. The views mechanism is a relatively new component of EDS and is still under development. It keeps a record of the alternative design possibilities being explored and the information that has been used. Thus it records the design process in terms of the alternatives tried and what they revealed (what was inferred from them). It also allows the user to organize the design task dynamically to suit the interests that he has and the exploration that he finds useful. Views are a means of modelling the structure of the designer's exploration at a higher level of organization that is seen from examining the dependency networks of the ATMS (Logan, 1992a).

A designer may try out several different design possibilities in parallel to see which one gives the best performance overall or to test the extent to

which the design specification is affected by (sensitive to) values of different design parameters. The generation of alternatives, however, is constrained by the choices which have been made in organizing and representing the domain (design) knowledge. In EDS the main restriction is imposed by the domain knowledge base. The choice of functional unit module taxonomy and the facets of the structured objects (properties and procedures) obviously constrains the designer to a particular conceptualisation of the artefact he is designing. Logan discusses this in the context of EDS and the broader issues of limitations of a model's view and its inherently value-laden nature through embodying a particular perspective of the design task in Logan, 1989. In this thesis the inherent limitations of models of expertise in particular are discussed in detail in chapter 4 section 1.

Evaluation of alternatives, viewed as determining the consequences of design decisions which define them, is carried out "automatically" by the system in the sense that consequences of decisions are inferred by the knowledge sources and the domain knowledge base. However, evaluation of them, in the broader sense, to determine what are the relative merits of alternatives, is left to the designer. The EDS supports the designer's decision making by providing information of the kind described in sections 2.3 and 2.6 above. The designer is supplied with information about alternatives so that he can evaluate the merits of one solution by comparison with others. When an inconsistency arises in a view or the design alternative that it represents is found to be unsatisfactory in some way the designer can address the problem directly or he can ignore it and proceed with development of the design alternative. EDS permits inconsistencies to exist within a view and will carry on, inferring new information based on what can consistently be derived from the view . By this means, the designer is further supported in evaluating alternatives relative to one another.

2.8 Support for Justification of Design Decisions

In EDS justification is defined formally for the nodes in the dependency network built and maintained by the ATMS. Each piece of information (a design decision made by the designer or a fact inferred by a knowledge source or the domain knowledge base) is associated with one of the ATMS nodes in the design description document. A piece of information is defined to be justified if a set of nodes from which it is derived can be traced. Assumptions, that is the design decisions taken by the user, are the ultimate authority for justification of a node i.e. the set of assumptions on which a piece of information, represented as a node, depends. The set of assumptions which a piece of information can be derived from is termed an environment. Environments are associated with labels so that a justifiable piece of information is tagged with the label or labels that support it. Unlabelled pieces of information are not justifiable and these are not used as the bases of new inferences. Justification of an inferred piece of data in EDS can therefore be seen as a conventional (first generation expert system)

trace back from consequent to antecedent until designer initiated assumptions are reached. The strength of EDS lies elsewhere, in its ability to allow inconsistencies to co-exist and in its ability to partition data into internally consistent sets of nodes. When new data is assumed or derived consistency checks are carried out, conflicts result in partitioning of assumptions into internally consistent sets and re-labelling of the affected pieces of data to associate them with the appropriate set of justifying assumptions.

EDS is able to provide these "neat" justifications because of the nature of the inferencing carried out by the general purpose knowledge sources. Their inferences are straight forward, e.g relational algebraic operations on tabular data, linear equation solving, and they do not interfere or overlap with one another. The inherent defeasible nature of the reasoning carried out by specialist knowledge sources simply through their inability to investigate or "know" the wider implications of what they infer is acknowledged, " ... the capabilities of specialists are finite. They are constrained to produce their advice based on the local problem context and cannot have knowledge of the consequences of their proposals for all design criteria in all situations." (Logan, 1992). Advice is always based on limited knowledge which can become invalid if the designer violates any of the assumptions of an advice-giving specialist on the basis of its limited view of the development of the design. The solution proposed for EDS is that knowledge sources should be seen to "initiate exploration" rather than advise in some more categorical sense, that is they should offer "reasonable suggestions" to the designer who is more able to judge their suitability in a wider context. However, it is not clear how this suggestion-only status of the system's contribution can be made obvious to the designer interacting with it. (In the description of Janus which follows in section 3 of this chapter and in particular in section 3.9, an interaction style and content deliberately intended to invoke in the user a suitable attitude to a support system's *suggestions* is described. The broader issue of revealing the limitations of a knowledge based system to a designer using it to support his design practice is dealt with in detail in chapter 5 in sections 2.1 and 2.3 particularly.).

Justifications of the designer's decisions are not recorded except in so far as they can be inferred from studying the record of the progress of the design which is represented in the design description document as the design history. In the "Design to Product" research programme provision was made for design notes to be recorded alongside the design specification but this facility is not documented in the published literature as a component of EDS.

2.9 Discussion

Two aspects of EDS are discussed further in this section, the nature of the design process it supports, and the way design knowledge is

represented.

2.9.1 Supporting exploratory design

The central elements of EDS are based on the belief that design problems are poorly, inconsistently and incompletely specified initially; that designers explore a design situation by trying out ideas (constructing partial, perhaps inconsistent solutions); and that designers may pursue a number of design possibilities in parallel, learning from evaluating each of them until finally, they arrive at a design solution which is satisfactory and possibly even good in comparison with the other feasible alternatives which have been exploited along the way. One of the most important contributions EDS makes is that it constitutes an engineered product which supports design on the basis of full acknowledgement of the wicked nature of design problems as far as they are currently understood. EDS accommodates an exploratory view of design through the organization of the blackboard and its consistency maintenance system. The consistency maintenance system by using a modified ATMS is able to support multiple inconsistent alternatives simultaneously. This allows the designer to move between views of the problem and different ways of approaching the task as he attempts to satisfy the design brief. This aspect of EDS sets it apart from much of the rest of the work of the A.I. in design community which either sets aside, ignores, or even denies what is known about the nature of designing.

It is interesting to note that recently initiated work to apply the exploration-based model of design and selected elements of the EDS architecture to the design of drugs which are sufficiently novel to be patentable seems to offer further scope for understanding and modelling the earliest stage of design. Smithers (1992) refers to this stage as problem formulation, others often refer to it as conceptual design, it is the same stage of design which is the focus of the case study in part 2 of this thesis.

2.9.2 Modelling design knowledge

The contribution of EDS to the way design knowledge is modelled is less clear. On one hand, it is claimed that the need to understand what goes on in design does not need to be so well understood when building systems which (just) support the designer (section 2.3 above). On the other hand, knowledge about how to apply domain knowledge to carry out design is claimed to be represented , at least to some degree, in the domain knowledge base. Design knowledge is described as knowledge about using domain knowledge "to define and solve problems: how the space of possible designs is explored and how a developing design problem structure is created, modified and refined" and is said to include "useful decomposition criteria and strategies, synthesis and analysis methods and techniques" (Logan, 1992b). The domain knowledge base is consistently described in

terms of two components, domain factual knowledge and the design knowledge described above. However, they are not seen as "orthogonal kinds of knowledge, there is often an important dependency relationship between domain knowledge and design knowledge which means that one cannot sensibly be expressed with out reference to the other" (Smithers, 1990, p.82).

In the early days of EDS when it was "put together" from previously developed research projects, the structure of the design knowledge base was largely determined by the functional unit module taxonomy (from VERIFY, Barrow, op.cit.) and by some input from a system called TROPIC (Latombe, 1976). In TROPIC three kinds of knowledge are distinguished; problem specific knowledge, domain knowledge about objects and concepts, and knowledge about how to select a problem-solving method for a problem. Design knowledge for specific tasks in product realization (in the "Design to Product" programme) was to be collected in separate specialist knowledge sources (support systems). Design knowledge would therefore be distributed among the components of EDS, some of it would be implicit in the component interactions. For example, "while the EDS shell provides a flexible environment for the definition of new support systems, such systems may violate some of the implicit assumptions on which the ATMS-blackboard model is based" (Logan, 1992, p.441). Further evidence that knowledge about the design task is encoded implicitly is given from the observation that designers when set the task of specifying the module classes for the domain knowledge base exploited knowledge of what the general and special purpose knowledge sources would do with them. This influenced them in deciding how the modules should be structured to achieve the desired effects.

In analyzing some of the short-comings of EDS Smithers (1989) identifies one of the main problems as lack of common understanding between EDS and the users about the task being carried out "the problem is how to provide the system with an understanding of what the designer is trying to do and how the designer is wanting to do it , so that it can better direct what it can do to support the task. It needs to have some kind of information about the purpose of the task and how the designer intends pursuing it" (Smithers, op.cit., pp.3-4). Smithers links to this point the issue of interaction between system and user for effective problem solving. The research in competent systems to which this thesis makes a contribution holds as central the claim that a shared model of what constitutes the task is essential between system and user if intelligible support is to be achieved ((e.g.) McDonnell, 1991,1991a). Smithers proposes that work should be carried out to understand how effective partnerships can be formed.

In the most recent work to apply ideas from EDS to support design of novel drugs which has already been mentioned (Smithers, 1992) explicit representation of the design task structure is evident. The (new style)

support knowledge sources are derived and organized on the basis of a decomposition of the task. Task analysis of the drug design domain has resulted in decomposition of the task into four main components. Each of these is further decomposed into sub-tasks. It is proposed that each of these will be represented as a knowledge source. Each knowledge source will be capable of achieving an explicit sub-task such as identifying the set of fragments of a molecular compound that deliver specific chemical properties. It is possible, therefore, to see EDS moving closer to a second generation expert systems architecture in terms of explicit representation of task structure and approach to the representation of domain and design knowledge.

3 Empowering Designers : Janus

For a number of years a group of researchers at the University of Colorado have been investigating how designers can be supported by what they call knowledge based design environments. A design environment is defined as a tool "that fosters human problem-domain communication by providing a set of building blocks that model a problem domain" and which incorporates knowledge about how the components can be fitted together (Fischer, 1989). The aims are to inform and support designers rather than to deskill or replace them. A number of themes are being explored but one essential, common element of their research philosophy is the idea of building "objects to think with". This has resulted in the development of a multifaceted architecture for design support. Attention here is centred on Janus, an application which makes use of the central components of the architecture to support the design of kitchen floor plans (Fischer, 1989, 1990a,1991, 1991a,1992).

3.1 Background

Two main relevant research themes can be seen in the evolution of Janus. One theme is centred upon the provision of intelligent support for designers using a cooperative problem-solving approach based on the idea that human-computer communication is best achieved by supporting "human problem domain communication". From this perspective, users are provided with “application oriented” building blocks with which to work. The idea is that designers, for example, should perceive design as communication with a domain of application rather than as the manipulation of symbols displayed by the computer. The aim is to make the computer “invisible” through the provision of relevant abstract operations and objects within the (computer based) design support environment. The operations and objects provide layers of abstraction above conventional (general purpose and low level) computer languages so that designers can work with primitives with which they feel comfortable (Fischer, 1989). In the case of kitchen floor plan design this means that designers are provided

with design objects which they can manipulate directly to form floor plan layouts.

The second relevant theme is directly linked to the research challenges proposed by Rittel (which concluded chapter 2), in particular to the idea that design can be viewed as an argumentative process. Since the early 1970s work on the support of design from this perspective has been undertaken under the title of issue-based information systems ((Kunz, 1970; McCall, 1989; McCall, 1991). Among other components, Janus contains a construction kit for creating a kitchen floor plan using domain oriented components, and a knowledge based critic for evaluating a design as it evolves. Like the researchers at Edinburgh, the builders of Janus believe that design is an ill-defined and wicked problem and that requirements fluctuate and may be conflicting. The researchers at Colorado believe that it is essential to design for the work people do rather than for an idealised description of it and in this respect they associate themselves with the work of researchers such as Ehn (Ehn, 1991), Bodker (Bodker, 1991) and Suchman (Suchman, 1987) whose work is usually "classified" as research into work-oriented human computer interaction. It follows from this perspective that design support (or design methodologies) based on the separation of problem setting and problem solving are not valued. Their view of cooperative problem solving is based on the idea that each participant (computer and human designer) should be encouraged to contribute what it is best able to do. They characterize humans as the creative partner, the one best able to set a task in its wider context, whilst the computer is seen as a dependable depository and manager of large amounts of information. They are concerned to remove tedious tasks from humans but not to replace their effort on tasks which they enjoy - such as "doing and deciding" (Fischer 1992 p.26).

The view of design on which the Janus system is based is one in which specification of the design requirements (problem setting) is *integrated* with constructing the design itself (problem solving). Design is seen to consist of two complementary activities which are termed constructive design and argumentative design. (Janus is so named because it has components which address these two faces of design.) Construction involves assembling a design solution using appropriate domain specific building blocks, e.g. for kitchen design these include cookers, sinks, cabinets and may include larger conceptual units like food preparation centres. Argumentation is the process of reasoning about the design and includes doing things like discussing a design problem with others or thinking about design principles. An example of a design principle (in Janus) is something like the work triangle in a kitchen - the distance along the sides of the triangle which links sink, cooker and refrigerator. Application of a design principle is open to interpretation as is the importance attached to it (Fischer, 1989).

3.2 Research Purposes

The objectives of the research are concerned with understanding how designers design and how they can be supported so that they can be more effective, be helped to avoid problems, and can learn new things about their design domain as they go along. The researchers apply their notions about design to themselves, as designers of design support systems. Therefore they are concerned to build computer based systems to test their theories and to learn from where they break down (cf. Schön's "reflection-in-action" and the ideas expressed by Lawson about designers as being solution-centred rather than problem centred chapter 2 section 3).

As has been noted above (section 3.1) there is a strong human-computer interaction theme in the research purposes, within this theme some work is concerned with learning how a computer system can be shaped into a useful and usable medium for design specialist in some domain such that they can get on with their jobs unhindered by the need to learn the languages of the computer (i.e. to get on and work in the way natural to them rather than having to learn to work on their problems in some "different" way that happens to be supported by the computer). Furthermore, it is an explicitly stated intention of the research to learn how to build interactive, knowledge based systems to cooperate in problem solving. The researchers deliberately distance themselves from the A.I. perspective which they characterize as an approach intending to understand and build autonomous, intelligent machines. They contrast this view with their approach which they describe as one intended to augment the creative and analytical skills of designers not to replacing them with automatic design systems. They also distance themselves from the idea that they are constructing expert systems giving as grounds that expert systems require complete understanding of a problem at the outset, something which cannot be provided in ill-defined problems such as design. "What has been made explicit always sets a limit, and there exists the potential for breakdowns that require a move beyond this limit." (Fischer, 1992, p.15) This comment is taken up in the discussion of Janus in section 3.9 of this chapter.

The main components of the Janus design environment are described below in section 3.4. It has been devised to support three main aspects of designing: reflection in action as defined by Schön, the evolutionary nature of designs (and thus of the design of the environments that support them), and the identification of information relevant at each step the designer makes i.e. making information ready to hand. (In this their view accords with a phenomenological view as described in chapter 2 section 4).

The view of design that is supported by Janus and by the other design environments related to it is summarised as follows,

> "Humans start from a partial specification, and refine it incrementally, on the basis of the feedback that they get from their environment. In designing, this feedback is provided by the 'back talk of the situation'. While engaging in a 'conversation with the design material', designers become aware of an occurrence of a breakdown. This awareness is triggered by evaluation and appreciation of the current design stage (artefact) in terms of the task at hand (goal). The evaluation is carried out by the designers themselves, or by outside agents (such as critics), in design environments. This reflection of the action results in the determination of a next move in problem setting and in problem solving." (Fischer, 1992, p.16)

It should be noted that in Janus reference to the design specification refers to specification of design requirements. The design itself is the design construction i.e. the kitchen floor plan designed. It is thus the design *construction* in Janus which corresponds to the design *specification* in EDS.

3.3 Roles of the Designer (User) and System in Interaction

In Janus the user is seen as an active agent who is in control of the design process supported by the system which participates actively in problem solving and decision making. The designer constructs his design by directly manipulating objects representing what he sees as components of the design. Components for designing with, known as design units, are selected by the user from a palette of icons which represent each design unit. Having selected a unit the designer moves it to a work area where the floor plan is to be constructed and places it in position. Changes to the design initiate messages (critics) which are about issues relating to the design as it currently stands. The designer is free to choose whether or not to explore the *argumentation* associated with critic messages in more detail and is also free to act upon or to ignore any information so revealed. By these means the user retains control of the design process, a critic interrupts only to draw the user's attention to potential for improving the design. (It is believed that critics can play a role in allowing the designer to learn about design issues as he designs.)

The designer also has available a catalogue of example designs which he can retrieve and incorporate into a new design. Thus he can reuse or redesign the floor plans (sample designs or previous designs of his own) stored in the catalogue. The design examples are catalogued according to their features (described further in the next section), these are dynamically organized to suit each design situation. As the designer works, a partial specification of the design requirements is built up at the same time as a partial design is constructed. Information from these sources is used to organize the catalogue so that the user can be helped to retrieve information *relevant* to what he is currently doing. The catalogue can be queried using

the partial specification and construction. The designer can refine queries to the catalogue by criticising the responses to earlier retrievals iteratively until he is satisfied with the design information supplied. This facility is provided by a retrieval-by-reformulation system called HELGON (Fisher, 1989a) which has been developed for supporting queries in situations where it is difficult for the user to be able to decide, without exploration, what a query should be. (Usually because the information stored is complex or because the user's task is not precisely clear.)

A designer can retrieve design examples that are similar to the floor plan which he is currently constructing by invoking a command to that effect. If he does this he is asked to specify the criteria of similarity that should be used e.g seek designs which use some of the same design unit types. This constitutes retrieval of examples on the basis of the current design construction. In this way users are gradually guided into refining the set of examples from the catalogue that are relevant to their current needs. A command to retrieve on the basis of the design specification (requirements) organizes the set of examples from the catalogue according to their relevance to the current design requirements specified.

When a designer has identified a design from the catalogue which he wishes to use he can either refer to it just to get ideas or he can use it to replace his current design construction (all or part of it) and then can continue to work on it further as necessary. It is for the designer to determine when a design is completed.

3.4 Main Architectural Components

Over the last few years a number of prototype design environments have been constructed. In that time the structure of the architecture has passed through a number of forms and new functionality has been added. The architecture described here is that reported as constituting the main elements of the Janus system in 1992. Examination of the literature shows a coherent progression from one engineered system to the next, each new piece of work addresses some of the central research issues which have been thrown up by earlier work. The main components of Janus are described briefly in two categories. In this section are described the (static) components which contain the domain knowledge and those which are associated with a particular design problem. In the next section are described the main (dynamic) mechanisms which manipulate and link the static components.

The static components can be seen by the user as five interface components namely: the construction kit including the current design construction (the design so far); a specification component; the catalogue of example designs; the issue base (the argumentation); and a simulation component. These are depicted in figure 3.3 which also shows the role of

each component in design support. (The simulation component allows the designer to see how the design will perform in use, it is not described further here.)

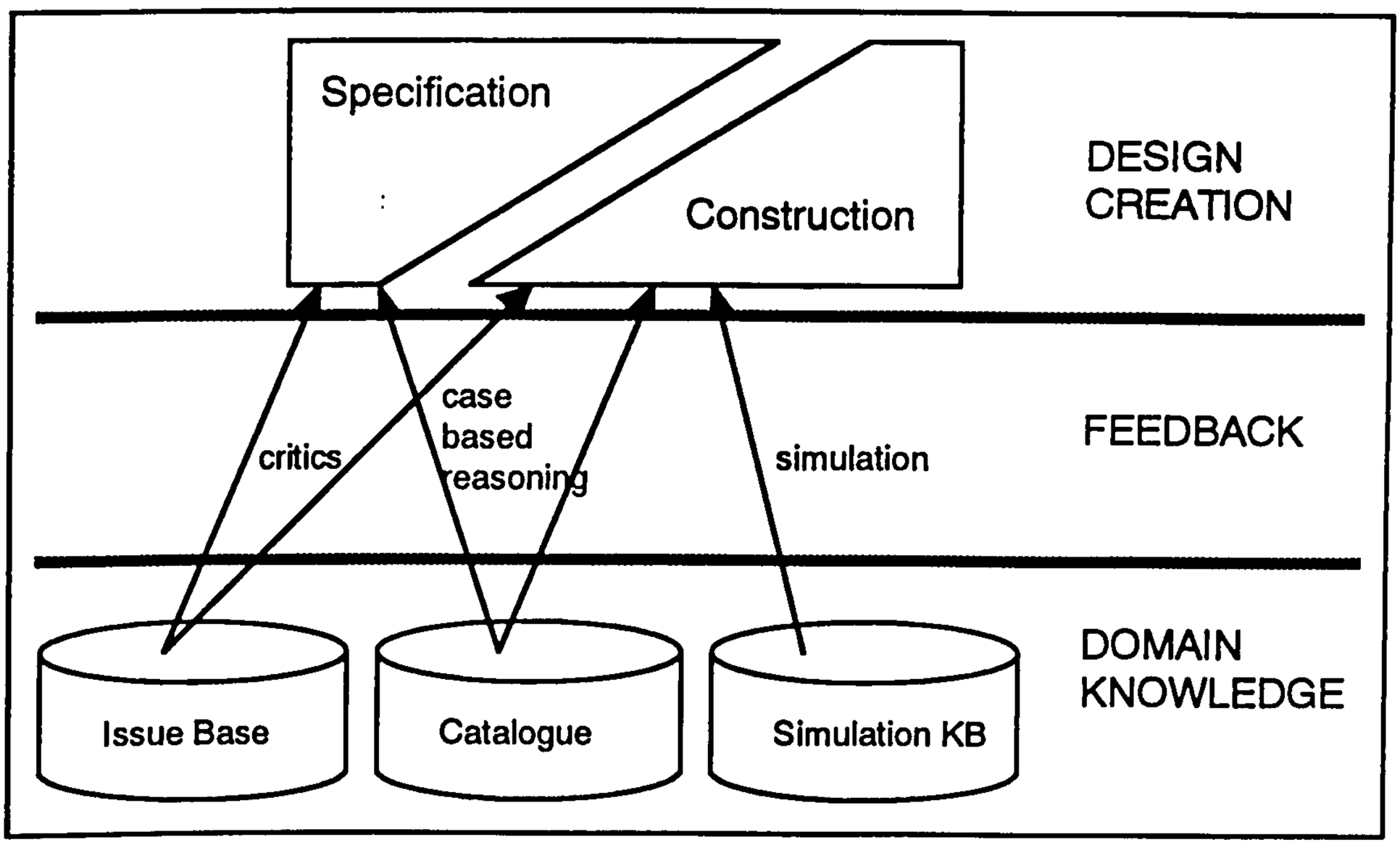


Figure 3.3 Main architectural components of Janus

Adapted from Fischer,G. & Nakakoji,K. "Beyond the macho approach of artificial intelligence: empower human designers- do not relace them", Knowledge Based Systems, Vol 5,No 1, 1992, p.17.

The two components associated with the creation of a particular design are the specification and construction components. As design proceeds the designer describes the required characteristics of the design (he specifies the design requirements) and indicates (by assigning weightings) the relative importance of these. As already mentioned, the specification of requirements, to the extent that they are known, play a part in prioritising and organising information presented from the catalogue of examples. A construction represents the constructed design so far in terms of the design units used and their relationship to one another. Each type of design unit has critic rules associated with it which , when a change to a design is made, can be triggered to invoke argumentation from the issue base. The construction kit seen by the user is a display of the construction so far and a palette from which design units can be selected. The palette constitutes the design vocabulary and as such determines the design space. Knowledge about how to combine the design units from the palette includes information about building regulations, safety standards and functional

preferences. Rules associated with building and safety codes are considered as fairly firm rules which should usually be observed whereas the functional preferences represent more subjective design practices which vary in value and importance from one designer to another.

Further domain knowledge is divided among the remaining three components. The catalogue has already been mentioned. It contains a collection of stored designs which can be reused directly in new designs or which the designer can consult for ideas. Objects stored in the catalogue are also used to illustrate critic messages and used thus play a part in warning of possible failures or weaknesses in a design. The issue base is said to capture the design rationale. Partial constructions have relevant issues linked to them via the critics, and the issues in turn are linked to one another. This is best illustrated by giving an example. If the designer places a design unit representing a cooker on the floor plan under construction in a position adjacent to a door a number of argumentation points may be brought to the designer's notice depending on the extent to which the designer decides to explore them (see 2.3 above). Pursuing this example specifically, points such as "cookers should be sited away from doors because of the fire hazard / burn hazard that they will otherwise constitute" and (on the other hand) "cookers near dining room doors are convenient for serving hot food" are issues that may be raised.

The components which have been outlined are the main components of Janus relevant here. It is noted, in passing, however that there are further components which have been developed to support other research work. One, for example, is concerned with designing and building design environments which users can modify to suit their particular design support requirements. A component of Janus, Janus Modifier, has been developed to explore this (Fischer, 1990). Description of this is not appropriate here, except to note that a designer may modify the critic rules in Janus to reflect his own views on the more subjective aspects of design practices.

3.5 System Dynamics

Users can approach design either by constructing a floor plan using the design units from the palette or they can design by modifying a design example taken from the catalogue. So, for example, users may commence design by stating a partial specification of the design requirements or by constructing a part of the design using the design units selected from the palette. When they cannot continue for some reason they may consult other components of the system for information. They may consult the catalogue which is organized for presentation to them according to relevance to the task at hand (as previously described). Alternatively they may explore the issues associated with the specification and construction so far.

Designers are able to move to and fro between the components of Janus gradually refining their understanding. Once a design has been completed it may be added to the set of examples stored in the catalogue. Starting with only a vague goal, the designer, aided by the system, gradually evolves both a specification of the design requirements and a constructed design. Along the way he is informed by the argumentation component (which uses the issue base), by relevant examples from the catalogue, and from feedback from the simulation component. The final product of this process is a specification of design requirements and a constructed design which meets these requirements. During design both the specification of requirements and the design construction are modified, the modification of either one having consequences for the other. Messages from critics are automatically invoked as additions and changes are made to the design. The designer can elect to attend to these messages and pursue them further by entering the issue base to explore the argumentation which underlies the message he has received. Once the user is attending to an issue he can explore related issues and arguments and see examples (the issue base is linked to the catalogue of examples) which illustrate the topics raised.

During construction of a design the user can initiate an analysis (a critique) of the partial design he has constructed so far. This analysis is based on the knowledge of design principles, both the firm ones about building and safety regulations, and the other more debatable ones. Violation of one of the rules associated with the design principles results in a message to the user and provides an opportunity to enter the issue base at the appropriate place to examine the argumentation associated with the potential problem. This mechanism is intended to support the reflection in action which is part of designing by signalling a potential breakdown to the designer (through the critics' messages) and then by supporting his reflection on it prompted by the argumentation in the issues base (cf. Merleau-Ponty, 1962, p.xiii). If an issue is rather abstract or conceptual in nature specific design examples which illustrate it are presented too, with the intention of helping the designer to understand the point being made.

Further support of the designer's reflection is provided by focussed access to the catalogue of stored examples. During design, the designer can make reference to the catalogue of stored examples using the mechanism for assisting exploration of *relevant* parts of the catalogue. The *catalogue explorer* retrieves design examples that are similar to the current design being constructed and organises the selected examples according to the design requirements as currently specified. Through this mechanism the designer is assisted in identifying and using information relevant to the task at hand. As the design changes, what is relevant to it, what is of interest to the designer, inevitably also changes. The identification of material which is relevant to the user at each step is achieved by dynamically selecting and organising what is presented to him from the catalogue. In this way the system attempts to provide a way of making information ready to hand, of moving the horizons, in effect to

acknowledge and support designing viewed phenomenologically.

3.6 Handling Design Trade-offs / Relaxing Constraints

Janus does not handle trade-offs autonomously, rather it supports the *designer's need* to do this in two main ways. Firstly, Janus does not constrain the designer to proceed forward monotonically towards a completed, refined design solution. The designer is free to add and modify both design requirements (the specification) and the designed floor plan (the construction). Secondly, the system alerts the designer, through the critics (the operation of which is described in detail in the next section), to parts of the construction that are inconsistent with the design requirements stated so far or with design principles. It is left for the designer to decide what action to take as a consequence of the "problems" highlighted. The system can assist the designer further, however, by judicious presentation of relevant examples from the catalogue (as previously described) and by presentation of argumentation related to the problem highlighted.

In Janus the critics do not resolve problems, their job is to recognize potential difficulties and to communicate these to the designer. (In some critiquing systems suggestions for overcoming problems are also made (Fischer 1991).) However, the fact that the argumentation includes arguments both for and against issues is intended to help the designer to assess the trade-offs to be made. If a piece of a floor plan construction violates constraints imposed by building codes and safety regulations, this is brought to the attention of the designer. Here again, decisions as to how to overcome constraints by modifying the design in some way, perhaps by making a trade-off against other requirements is left to the designer. The designer is provided with a facility by which he can assign weightings to the design requirements he has specified so that the system can rank their relative importance. This information is used to overcome dilemmas of specifications which conflict with one another and is used, for example, in organising design examples retrieved from the catalogue for presentation to the designer when requested. Thus, it is left to the user to make the kinds of trade-off associated with prioritising specifications, but having done this, and communicated such priorities, the system makes use of them to further support the designer.

3.7 Generation and Evaluation of Alternatives

Generation and evaluation of a design are explicitly interwoven in Janus as a result of the view of the design process which it supports - that design involves the two complementary activities of construction and argumentation. Generation of the design and of its associated requirements specification are achieved using the components for construction which have been described in sections 3.4 and 3.5 above. Whilst Janus itself does not directly support parallel development of a number of alternatives

simultaneously as EDS does, it is possible for the designer to construct and store a number of designs (by adding them to the catalogue of examples) and thus to compare a number of completed or partial design alternatives. The catalogue can also be used to generate design alternatives through reuse and adaption of the examples it contains. The catalogue is not restricted to containing examples of good designs, it offers the potential for designers to learn about design from failures and designs which are poor in some respects.

To support consideration of alternative possibilities at a decision point during design construction the user can refer to the issue base for information about the pros and cons of particular choices. The designer can explore further by following links to related issues in the issue base.

Janus offers powerful support for evaluation of designs both in terms of automated evaluation by the system and in the support it gives to the designer to evaluate the design for himself. This support is provided through the adoption of a critiquing approach to problem solving. In Janus the designer constructs designs (or partial designs) and the critiquing system analyses them to provide a critique for the designer to consider. The designer then (optionally) revises his design , it is criticised again, and so on until the designer is satisfied with the design. Research work on issue based information systems predating Janus showed that if designers are provided with a purely passive argumentative component they tend to work on design construction and make little reference to the issue base. In Janus, therefore, critics operate as active agents during construction, alerting the designer to the availability of information and issues relevant to what he is currently doing (constructing). The active critics in Janus use rules which are mainly concerned with spatial relationships between design units. The critics detect inferior aspects of a design and through their links to the issue base they provide explanations and argumentation for their criticism. The way in which construction triggers a critic and may lead to the designer attending to some relevant issues in the issue base is best illustrated by paraphrasing an example concerning the siting of a cooker given by Fischer (Fischer, 1991, p.704):

> During design the designer moves the cooker within the floor plan. This triggers the cooker critic which tests the location of the cooker relative to the doors, the sink, and the refrigerator which are currently part of the constructed design. Messages from the critic are displayed which tell the designer that the cooker is not well located with respect to the door and the sink and that the work triangle (subtended between cooker, sink and refrigerator) exceeds the distance recorded as good design practice. A problem is thus communicated to the designer. He is prompted to give it some consideration (to reflect on it). A kitchen safety rule has been broken by the fact that the cooker is too near the kitchen door.

The designer may not have known this safety rule or may not understand the reason for it, in which case he can seek explanation. Janus does not display stored text giving one view. It attempts to present several perspectives. When the designer indicates that elaboration on the message about the safety rule is wanted the system invokes the argumentation facilities , the message gives the argumentative context, i.e. the issue being raised, which concerns where a cooker should be placed in a kitchen. The issue base shows how a cooker should be positioned in relation to a door and gives the arguments for and against this. (These have been given already in section 3.4 above.) An example of how positioning a cooker successfully in relation to a door has been achieved in an example from the catalogue is displayed as a concrete illustration. The designer can continue to explore the issue by navigating around the issue base (using the hypertext style of interaction which the implementation supports). When the designer has satisfied his information needs he returns to the construction of his design and makes any alterations that he sees fit.

The use of a critiquing approach for the generation and evaluation of designing is appealing from the view that generation and evaluation of designs proceed hand-in-hand. The critiquing approach raises many research issues of its own. Among these are debates about whether critics should praise as well as offer negative comments, to what extent they should advise on problems as well as detect them, when they should intervene, how often, and to what extent they should be adaptable and / or adaptive (Fischer, 1990a, 1991). The wider issues associated with critiquing are beyond the description of Janus presented here (which is restricted to the critiquing currently supported by the Janus system). The matter of supporting critics is returned to in the discussion of Janus in section 3.9 of this chapter.

3.8 Support for Justification of Design Decisions

The argumentation supported by the issues base is the main source of information for the designer to use in justifying his decisions. The critics are able to explain the reasons for their messages having been invoked by reference to the issue base. The argumentation associated with a critiquing message is intended to help the user to understand (or even learn) the design principles which apply, the argumentation shows why they apply and when they can validly be ignored or overruled.

> "Critics have to be able to explain the reasons for their interventions. This provides users with the opportunity to assess the critique and then to decide whether to accept it." (Fischer, 1991)

Designers are supported in justifying their design decisions in a manner which is consistent with the way they are supported in design, designers are supported in such a way that *they* see the wider context. Janus' designers' stated belief is that one of the human designer's strengths and one of a computer system's weaknesses is to be able to set a task within a wider context. This ability, an essential element of justification, is discussed in detail in chapters 4 and 5. In Janus the designer is unambiguously always in control and is always the partner in design who does the deciding and with whom the responsibility is seen to rest. He can make use of information from the issue base not only for deciding about the construction of his design but also to support the case for what he produces.

Further indirect support for justification can be provided from the catalogue of design examples. A designer can make use of the catalogue to show what the consequences of alternative decisions might be, he is helped in this through the ability of Janus to identify examples from the catalogue which meet not only the surface features (determined by the structure of the design) but also hidden features of the design (properties like safety level). Some hidden features can easily be quantified such as the requirement for the kitchen to be less than a certain total floor area. Other hidden features are more subtle, they are subjective ones like deciding what importance to assign to safety. Domain knowledge in the form of rules links subjective hidden features of a design specification to examples in the catalogue of pre-stored designs. When design examples are retrieved for the designer to consult these rules are used, as well as more straightforward matching requirements (e.g. that the kitchen must include a microwave oven), to retrieve relevant designs. Janus produces the specification linking rules in a non-trivial fashion by deriving them dynamically from the contents of the issue base in which arguments are represented in a formal way. This permits the necessary values of physical features of catalogue examples which satisfy hidden feature specifications to be inferred (Fischer, 1992).

3.9 Discussion

Two aspects of Janus are discussed further in this section, firstly, the way in which the knowledge of the design support system is presented so as to make its limitations clear to the designer, and secondly, the limitations of a critiquing system in giving guidance to a designer.

3.9.1 Revealing limitations

Critiquing and argumentation are used in Janus to reveal both the shortcomings of the designer (in terms of what he has designed) and also something of the limitations of the knowledge bases which are supporting him. The strength of the critiquing approach seems to lie in the opportunity it presents for tightly coupling the generation of a design with

the evaluation of it. In Janus this is achieved by supporting construction with prompts in the form of critic messages suggesting points at which the designer may benefit from consulting the argumentation in the issues base. The critiquing components are intended to "talk back" to the designer by signalling shortcomings in the design, "by showing that the artefact under construction has shortcomings, critics cause users to pause for a moment, to reflect on the situation and to apply new knowledge to the problem as well as to explore alternative designs" (Fischer, 1991, p.717). In this way they support the notion that professional practice is both action and reflection. The critiquing components, namely, the critics themselves, the issues base, and the catalogue of examples which are used to give concrete illustrations of the abstract arguments play an important part in alerting the designer to potential problems with his design. They bring relevant issues to his attention and reveal the basis for the critics' advice. In this last role the fact that several (possibly conflicting) aspects of an argument (different points of view, different design priorities) are presented is a vital one. The system as a whole, and the philosophical background within which it has been designed, firmly places responsibility for all design decisions with the designer. This stance is reinforced by the style of interaction and the way in which the issues are represented and presented to the designer in an argumentative manner.

3.9.2 Critiquing's limits on guidance

Whilst a strength of the Janus system is the way in which responsibility is "handed back" to the designer, there is a major weakness in the critiquing system when it comes to the ability to give the designer guidance. In Janus there is no explicit representation of the task the designer is undertaking, the critiquing components have no information about the designer's purposes or goals. This limits the amount of direction that the system can provide even though as much information about what is relevant as possible is extracted from the context provided by the partial design constructed and the design requirements as the designer has stated them so far. (Considerable sophistication is at times employed in the methods to do this - as described in the previous sections above.) The designers of Janus acknowledge this limitation. "Supporting users in their own doing means that details of user goals are often not available to the system, limiting the specificity of the critique the system can provide" (Fischer, 1990a, p.346). Their approach to overcoming this limitation is to explore critiquing among humans and to try to model that more closely. At present most critiquing systems (just) respond to the user's actions by making suggestions, explaining points and providing supporting arguments. The designers of Janus have observed, however, that "human critiquing is a more cooperative problem solving activity, during which an increased understanding of the problem develops". It will be seen that the importance of representing the task structure in models which form the basis of design support systems is a major issue in chapters 4 and 5.

4 Review

Each of the three systems described in detail in this chapter represent significantly different approaches to design support. The differences in perspectives have been emphasised by including in the descriptions of each system explanations of the theoretical stances and the larger research purposes of the groups of researchers responsible for each of the systems. Each of the systems has been linked to some view of designing expressed in terms of the ideas about designing reviewed in chapter 2. By describing these three systems in detail it has been possible to introduce informally the main issues which are analyzed and developed formally in chapters 4 and 5 of this thesis.

CHAPTER 4

Modelling Designers' Competence

"What, in fact , is the absurd man? ... he prefers his courage and his reasoning. The first teaches him to live without appeal and to get along with what he has; the second informs him of his limits."

Albert Camus

This chapter begins with a discussion of important general issues associated with the modelling of expertise (section 1). Attention is then focused on describing the modelling capabilities of second generation knowledge based systems based on a Competence Model (Keravnou, 1986; Johnson, 1986; Johnson, 1986a). Research into developing Competent Systems based on models of competence has occupied researchers at Brunel University for a number of years. Work to develop a knowledge based systems architecture, first applied to fault diagnosis (Keravnou, op.cit.), has been followed by related studies extending the original ideas in the direction of intelligent data handling and distributed knowledge bases among others (Murdoch, 1990; Stylianou, 1991). The broader methodological aspects of competence modelling have been pursued particularly in the area of knowledge elicitation (for example Graham,1990; Tomlinson, 1993; Funes, 1994). The methodological aspects of eliciting a model of competence are an important aspect of the work presented in this thesis but these matters are dealt with in chapter 6. In this chapter description of the Competent Systems architecture is confined to presentation of an interpretation of those aspects which are most relevant to supporting design activities (section 2).

The issues of concern for applying the general notions about modelling competence to the modelling of engineering designers' competence specifically are introduced in section 3.1. The main themes arising from the shortcomings of knowledge based systems for design are reviewed in section 3.2. These set the scene for a more detailed description of the implications for knowledge based systems architectures capable of supporting a designer's abilities to make trade-offs among conflicting requirements, to generate and evaluate alternative design ideas, and finally to justify their design decisions. Each of these aspects of designers' behaviour are considered separately in some detail in sections 3.3, 3.4 and 3.5. The chapter concludes by arguing that the Competent Systems architecture offers a basis for supporting these aspects of designers' behaviour.

1 Modelling Expertise

In this first section of this chapter issues concerning the modelling of expertise are addressed. The nature of expertise is summarised (section 1.2) and the way in which it is modelled in second generation expert systems is introduced (section 1.3). Some important aspects of models in general and of models of expertise in particular are set forth (section 1.4). This part of the chapter concludes with a discussion of the notion of "deep" knowledge as it has been applied to knowledge based systems technology (section 1.5).

1.1 Knowledge Based Systems as Models

The building of knowledge based systems is (itself) a creative design activity in which a systematic domain is created covering certain aspects of a professional's work (Winograd, 1986, p.175). Competent professionals have learned effective strategies and heuristics which make them capable of applying their knowledge productively in a particular domain. They become expert through study, training, and above all through practical experience in their chosen profession. It is possible to model competence to varying degrees of adequacy in a knowledge based system. First generation expert systems used a knowledge representation which promoted the building of models in which the different parts of the knowledge base bore no structural relationships to the uses to which they were put. Evaluation of the performance of these systems tended to be single-dimensional - in terms of their ability to give the "right" answer. Second generation expert systems use knowledge representations which produce richer models of expertise. They employ knowledge structures which explicitly represent components of experts' knowledge and the uses of that knowledge. They attempt to model competence. One of the main motivations for their development has been a desire to move away from simplistic performance evaluation, towards the construction of systems which the user perceives to behave logically (intelligently) and which can explain their conclusions.

1.2 The Nature of Expertise

Novices in a profession tend to use an unstructured collection of factual knowledge to solve the problems with which they are presented (Feltovich, 1984; Brown, 1981). For instance, inexperienced designers use experience of physically similar situations rather than the more powerful and flexible conceptual constructs which they develop as a result of increased experience (Akin, 1988). As experience is gained in a chosen profession, factual knowledge becomes structured in a way which reflects the strategies and the heuristics that are adopted by the practitioner to tackle usefully the professional challenges with which he is faced. Thus a designer learns how to approach problems in a way which matches his

capabilities (Akin, op.cit.). Putting this in more general terms, the way in which the knowledge of the competent professional is applied is reflected in its structure (Keravnou, 1986). Thus, knowledge becomes structured through experience. The structures formed assist the professional to manage and use effectively knowledge for tasks which are demanding through their complexity and inherent uncertainty. Expert professionals are seen to use "artful ways" to deal competently with the indeterminacies and value conflicts of practice. They usually know more than they can say[1], Schön (Schön,1983) observes that, in fact, professionals are disturbed "to find that they cannot account for the processes they have come to see as central to professional competence" (op.cit., p.19).

Individual professionals do not work in isolation cut off from interaction with others. Their knowledge is therefore organized for effective problem solving (use) in a way which allows communication and co-operation with others. In design, this interaction takes place both within the design team and with outsiders. A designer's need to explain his reasoning and to justify decisions is an integral part of the practice of his profession. This need is supported by the way his knowledge is structured. The importance of communication in design is indicated by its status as one of four primary phases in design as specified in the RIBA handbook on Architectural Practice and Management (RIBA, 1965)[2].

"Application" of expertise cannot therefore be legitimately separated from its communication, and the organizational setting of the task to which it is applied. Action (decision making) is always predicated by context and purpose, and is not intelligible in isolation from these. Expertise is the application of knowledge within a social context and this context embodies standards and criteria of acceptability which constitute the perspective within which professional choices take place and justifications are made. It is the contextual framework itself which determines the validity of the expertise[3]. Rowe (1987, p.37) has observed that design is to be seen as a normative enterprise - design proposals are about

---

[1] An every day example is that of proficiency in a language - it is not dependent on knowledge of the processes that generate utterances but we can model a speaker using productions and phonemes.

[2]The four phases, which are not to be viewed as sequential, are designated ; assimilation, general study, development, and communication.

[3]The calling to mind of the ideas of structural linguistics which occurs here is no coincidence. The distinctions between language (langue) and speech (parole) and their relationship to one another gives the basis for, and determines the value of, the use of systemic grammar networks for representing aspects of competence. The end note to chapter 6 elaborates on this point.

what is proper. "Practice has to do with others and co-determines the communal concern by its doing" (Gadamer, 1981, p.82).

Where a complex task is carried out in some organizational setting, the expertise needed may be distributed over many individuals working co-operatively. For example, although invention by individuals is still possible, design and re-design of products is now seldom carried out by an individual. There may be one individual who inspires and leads but he is dependent on the contributions from, and co-operation of others, of the team for the quality of the product (Flurscheim, 1977). Knowledge of the structure of a task within an organizational environment is an essential component in the understanding of an individual's role , the issues that arise, and the argument - the discourse - that takes place. Thus, a competence model does not have to be elicited from one individual. A competence model only has to model what can be done in a domain. It is not necessary to think of it as a model of the "cognitive processes" of an individual (Keravnou, 1986). The model is more like a grammar of a language than the psychology of a speaker. In knowledge acquisition, professionals from a domain "work together with knowledge engineers to articulate the relevant concepts" and to model how they are used. The model is a "formal representation that deals with things that the professional already knows how to work with, providing for precise and unambiguous description and manipulation. The critical issue is its correspondence to a domain that is ready to hand for those that will use it" (Winograd , op.cit., p.176).

Systems which purport to model competence must make explicit the domain knowledge (how it is organized and of what it consists), the tasks, the strategies, and the heuristics adopted. The validity of the advice offered by a knowledge based system is clearly important, but is not sufficient, in itself, to provide a technology which will be accepted and used. This is why it is important to move away from systems that rely on rules which treat knowledge in a uniform manner because these systems do not distinguish different types of knowledge. If a knowledge based system is to model competence, the structure of the task and the factual and strategic knowledge must be distinguished and represented explicitly. Strategic knowledge must be shown to be exercised through the structure of the factual knowledge of the domain. Knowledge engineers, then have to take a disciplined approach to create structural descriptions of knowledge, driven by the specific characteristics of a domain and of expertise, which are expressive enough to reflect the components of a "knowledge grammar".

1.3 How Competence is Modelled

Competent professionals do not obstinately pursue the same set of actions, in the same sequence, for every task with which they are faced. They make choices both of what to do (where to focus their attention, what part of a task to attempt), how to do it, and what strategy to use, depending

on the circumstances with which they are presented. It is clear that an overall task, and parts of it, can be achieved in a number of ways. Competent operators choose appropriate strategies to fit the circumstances, and in this way they display their flexibility in solving problems or completing tasks. Designers tend to adopt strategies which will lead them to learn more about the design situation they are faced with and which as a result will lead them towards an acceptable design solution (Lawson, 1990, p.135). The way in which experts *actually* go about their task in a specific circumstance is dictated by the particular facts which pertain. For example, designers, observed in action by Visser (reported in chapter 2), behave opportunistically to make efficient use of resources.

In a model of competence which forms the basis of a knowledge based system based on the Competent Systems architecture, the dynamic aspect of the model (what is done) is achieved through movement within the task structure which represents what can be done. What an expert actually does is related the set of possibilities available to him, so in a model of competent behaviour the *dynamic* operation is clearly related to the *static* task structure. In a knowledge based system the static structure represents what can be done i.e. the tasks that can be attempted. Representation of strategic knowledge is achieved by explicitly encoding the choices (strategies) available to achieve each part of the overall task and the basis on which choice from among them is to be made. In the next section of this chapter (section 2) the static and dynamic modelling capabilities of the Competent Systems architecture (Keravnou, 1986) are described in detail. In this architecture task structure, strategies and conditions for strategic choices are explicitly represented. Reasoning is represented by a sequence of (sub-task) instantiations and is controlled by the circumstance-specific data satisfying conditions for choosing which step to take (i.e. which sub-task to instantiate) next.

1.4 The Nature of Models

The relation between a model and what it models is one of analogy (Hesse, 1974). A model is not assumed to exactly describe a real system but to be analogous to it in some important respects. Models do not characterize real systems with complete accuracy, however they are intended to be useful within a narrow range of application. A modelling claim has the form - "This given, real system is very much like the kind of systems defined by this particular model in certain respects". This is a weaker claim than one of isomorphism between a real system and systems defined by a model. Systems defined by a model will be like real systems in some respects, Hesse calls this the "positive analogy". In some other respects real systems are not like the systems defined by a model, this gives the "negative analogy". All models have some negative analogy to the systems they are intended to model. A modelling claim applies only to the intended

positive analogy and ignores the expected discrepancy between the real systems and the model defined systems[4].

When the phenomenon to be modelled is competence, there are further implications to be considered. A well-grounded competence model is inherently limited. Firstly because, by definition, it models from a perspective, for a purpose, in a context. But secondly, in this case, because what is modelled is inherently defeasible. It follows from this second aspect that revising the model by redefining or extending its perspective and purpose - whilst this may result in a model that covers "more cases" - will never produce a model whose conclusions are not defeasible by some premise not modelled. It is this aspect of the relationship of the model to what is modelled in knowledge based systems which gives rise to important implications for the role of interactions between the user and the system. Three purposes, or a three-fold role, can be distinguished. The first is to show the model's working, and hence (the second) to make a modelling claim. The third aspect of the role is to make apparent the discrepancy between the real system and the model defined systems which the knowledge based system embodies. This three-fold role is largely discharged through the explanations given, but it is not achieved by distinctly separate explanation facilities within the knowledge based system. If a knowledge based system is to be useful in supporting the practice of a profession the inherent defeasibility of the model on which is based must be faced. This issue underlies all of the discussion of requirements for supporting design practice in chapter 5 (section 2).

1.5 The Contribution from "Deep" Models

Attempts have been made to improve expert systems by making them "deep" (Steels, 1988; Chandresekaran, 1991). From this perspective a model is "deeper" when the knowledge or heuristics are decomposed into what some have called “expertise-neutral” components which form causal, structural or physical models. These models are said to underpin the expertise. It may well be the case that in particular fields of expertise experts routinely have access to “deeper” models in this sense and in these cases a model of expertise will have embedded in it a "deeper" model. However, to acknowledge this is not to say that the expertise has been “decomposed” into a deep model. That would be to fall prey to the decomposition fallacy[5]. In professions where experts have access to causal

---

[4]However it is not true that what is not and cannot be represented in the positive analogy is denied by the model (Ryle, 1954, p.83).

[5]The complex concepts, the view of the task, the strategies and the knowledge structures which an expert employs in order to operate competently are arrived at through recognised training and prolonged practical experience. It does not necessarily follow that

and physical models and routinely reason with them, there clearly a model of expertise is improved by representation of causal structures and reasoning associated with them. Expressed in terms of modelling claims, where such deep models are validly incorporated, the effect of their inclusion is that more detail about the positive analogy aspects of the model can be given. However, if the rationale for the expert system as a model of competence is to be adhered to, this approach must be managed cautiously, for the granularity and structure of the knowledge modelled must be dictated by what is used in competent performance.

If "deeper" models include representation of structures that do not reflect what experts actually do then the "expert systems" which are based on these models cease to belong among that class of systems which can claim to be models of expertise. Designers of knowledge based systems must keep in mind their starting assumptions and must take care not to violate the premises on which the modelling is based. Expert systems which contain components which are not structurally related to components of expertise inevitably incur consequential effects on the nature of the interaction they can support and the relevance of the explanations they can give. The link between a model of competence and the expert, through which interaction, including explanation, makes sense relies upon a coherent interpretation of the task and of the professional norms which apply.

The fundamental issue which underlies all of this concerns the nature of "practical" reasoning. On the one hand there is the notion that it is means-ends reasoning within a framework of scientific knowledge, and hence the notion of technology as application of scientific knowledge. On the other hand, there is the tradition which holds that practical reasoning is not a species of scientific knowledge but is a fundamental kind of reasoning[6] with its own purpose and value which consequently cannot be reduced to means-ends reasoning (Johnson, 1991). From this latter view of practical reasoning, knowledge is implicit in patterns of action (Gadamer, 1981; Ricoeur, 1981; Ryle, 1949; Schön, 1983).

If professional practice is based on the tradition of practical reasoning as something undertaken by social beings, rather than being abstract

---

these concepts, given the nature of their complexity, can be simply decomposed into somehow more manageable elements without loss of some essential specific characteristics of the complex phenomena (the expertise) from which the decomposition is derived (Ryle,1954).

[6]From this perspective on practical reasoning, the following notions have been challenged: technical rationality; the application of "scientific knowledge" to solve problems (Gadamer 1981); and the reduction of "knowing how" to "knowing that" Ryle (1949).

deductions made within some formal framework of scientific knowledge, then there are inescapable consequences for those building systems to support that practice. These consequences are addressed in more detail in chapter 5. Here, to conclude this discussion, it is sufficient to observe that given that competent professionals are social beings their competence needs to be modelled with due recognition of its fundamentally social characteristics. The interaction enabled by a model of competence which forms the basis of a knowledge based practice support system inevitably reinforces the commitments being made in the technology which supports it.

2 Competent Systems

In this section components of a second generation knowledge based systems architecture for modelling competence (referred to as the Competent Systems architecture) are described. The architecture has evolved and continues to do so within a methodological framework which takes into account the theoretical issues which have been raised and discussed so far in this chapter. Description of the architecture is confined to that which is most relevant to supporting the aspects of engineering designers' behaviour on which this thesis focuses.

To be consistent with what has been discussed above (in section 1), competent designers are to be seen as those who have, through experience, structured their factual knowledge of a particular design domain in an effective and efficient way for the purpose of carrying out the design activity at which they are deemed to be expert. In a model of competence which is to form the basis for a knowledge based system the structural organization of the domain knowledge is made explicit as is the strategic knowledge which enables the expert designer to operate effectively, to make use of what is specific to the current design task, and to direct progress towards a satisfactory design solution.

In a knowledge based system based on a model of competence explicitness is achieved through identifiable computational structures. The computational structures are the level of abstraction that constitutes the architecture of the system. Hence, it is through the architecture that the modelling capabilities, both static and dynamic, are facilitated. The Competent Systems architecture offers the computational structures needed to support a model of competence, the major aspects of the architecture i.e the explicit task structure with data driven execution of its sub-tasks and the static and dynamic aspects of the modelling capabilities are described below in the remainder of section 2. The description in section 2 refers to components of the Competent Systems architecture originally developed by Keravnou and Johnson (Keravnou, 1986). In section 3 discussion becomes design specific as the characteristic features of design

competence are discussed and the potential of the Competent Systems architecture for modelling them is explained.

2.1 Explicit Task structure and Data Driven Task Execution

In common with some other second generation architectures, the Competent Systems architecture supports explicit representation of the task structure. Models which include this have the potential for giving an explanation of why each step has been taken in terms of the task itself. The ability to provide explanations from a task structure has long been considered to be richer and generally more satisfactory than that which is possible from the kind of trace provided by systems based on independent rules which contain no such explicit representation (Clancey, 1983; Mostow, 1985). Making the task structure explicit in the representation of the competence, provided it accurately models the human expert's approach, makes possible the provision of empathetic explanations, ones which the knowledge based system user can make sense of, and thus provides the basis for a partnership between the user and the system in which man and machine co-operate through a shared, common model of the design process (Cohen, 1987; Mostow, 1985).

Considered statically, the explicitly represented task structure captures what competent professionals can do and the structure of what they can do in terms of decomposition into possible sub-tasks. The way in which experts actually go about their overall task in a particular situation is dictated by the information they have to hand reflecting the particular circumstance in which they are using their expertise. The movement through the task structure in a model of competence is supported by the dynamic modelling capabilities of the knowledge based systems architecture as described below (section 2.2). Competent behaviour (the dynamics) is clearly related to the static task structure i.e. what an expert actually does is related to the set of possibilities available to him (as already stated). This is what the static structure represents.

In the Competent Systems architecture the (sub) tasks which comprise the (overall) task structure are represented declaratively and are treated as data which are interpreted by a task interpreter. The function of the task interpreter is to create an instantiation of a task, to execute the task, and to link task instantiations to form a trace of the dynamic progress towards, in the case of a design support system, a design solution. The trace records tasks executed and is a resource for explanations, system debugging, and for reasoning about the design decisions which have been made. The task interpreter is called recursively to execute sub-tasks. The sequence of task instantiations representing movement through the task structure in a particular case is determined by the situation-specific data since the (sub) tasks identified in a task structure may be carried out

in sequence or as dictated by a set of strategies which are explicitly represented as possible means of achieving each (sub) task. The logical bases for selecting a particular strategy to achieve a part of the overall task are explicitly represented as conditions which must hold for a strategy to be enabled or disabled. The conditions for achieving a (sub) task using a particular strategy refer to the situation specific data i.e. the current state of evolution of the design in the case of a design support system. The architecture allows for the designers' ability to be flexible - to decide what to do on the basis of what applies in a particular situation. It does this through supporting representation of appropriate strategies to achieve a (sub) task based on an understanding of the logical bases for choosing among the strategies available (Johnson, 1986, 1986a). In a system based on a competence model the sequence of (sub) task instantiations representing the movement through the task structure is driven by the problem specific data.

2.2 Static and Dynamic Modelling Capabilities

In much of the literature describing existing knowledge based systems architectures there is a tendency to consider the task decomposition (explicit or implicit) as a goal tree. One static structure represents both the task structure and the possible access paths to each sub-task. The structure therefore represents both a task tree and a search tree. Where a strictly hierarchical goal structure represents the successive decomposition of tasks (seen as goals) into sub-tasks, the implicit assumption is that search proceeds following an "establish and refine" procedure. In such an architecture explanations of the "how" and "why" varieties (described below) are given by reference to the goal structure and consist of displaying each inference step. If the task decomposition is based on valid analysis of the expert's behaviour, the explanations will appear to be relevant and comprehensible to him. The dynamic modelling capabilities of knowledge based systems with this sort of architecture can be understood directly from examination of the static goal structure.

In the Competent Systems architecture the task structure is represented statically by an explicit task tree which represents the actual task decomposition. As with other architectures which support explicit representation of tasks, the task structure provides the potential for explanations which address the purpose of, or set in context in some way, a particular action of the system. As intimated above, explanations made on this basis are considered to be meaningful to the user provided that the task structure models the user's perceptions of the activity. However, the rich modelling potential of the Competent Systems architecture is revealed when its dynamic operation, its operation during a particular design task, is examined. Dynamically, the Competent Systems architecture permits movement over the task tree to take the form

of a graph search, directed by the data currently relating to the design. Thus, statically, task decomposition is tree-like but access to tasks is not enabled solely through the static route provided by the task tree acting as a search tree. Additional routes are provided through the strategies associated with each non-leaf node in the task tree which can direct movement within the task tree via enabling, disabling and relaxation conditions which determine which (sub) tasks to attempt next on the basis of the current state of the design and the history of what has so far been attempted (which sub-tasks (nodes) have been visited, and under what circumstances). It can therefore be seen that a path to completion of a design cannot simply be read off the static task structure representation.

A knowledge based system which has explicit representation of the task structure and of the strategies which are available to effect movement through it has the potential to provide explanations to "why?", "how?" and "why not?" questions. Examples of these are "why are you asking that?", "how did you conclude that?", "why did you not try that?". More simply structured systems which lack explicit representation of strategies and the bases for choosing them such as those constructed from independent rules cannot give "why not?" explanations and their responses to "why?" and "how?" enquiries refer to the rule interactions which have been effected which do not necessarily model the reasoning of an expert (Clancey, 1983).

In the Competent Systems architecture "why" explanations are concerned with why a particular task instantiation was created. An explanation is given which includes abstract elements provided by the explicit task structure and concrete elements which explain the particular circumstances (the data) giving rise to the task instantiation. The explanation facility makes use of the abstract conditions explicitly encoded which represent choice conditions for the strategies which determine the (sub) task instantiations. The "how" type explanations use the same resources to describe how a particular (sub) task instantiation has been accomplished. In the context of a particular "why" type explanation the reasons why alternatives were not pursued can be requested through a "why not" explanation which describes in abstract terms why a particular alternative (sub) task or strategy was not selected when it was an option at a decision point.

Having described the main components of the Competent Systems architecture generally relevant for modelling competence, in the next section of this chapter attention is focussed on characterising the competence of engineering designers specifically.

3 Modelling Engineering Designers' Competence

Engineering designers make use of technical expertise relevant to their discipline and tend to approach design in an organised way. For example, a typical highly systematic approach to engineering design favoured by many engineers in Germany for both the teaching of engineering design and the practice of it is described in some detail by Pahl and Beitz (Pahl, 1988). Throughout design, technical knowledge, the body of knowledge associated with the domain within which the design falls, is applied. A competent engineering designer, like his fellow competent professionals in other disciplines, has structured his technical knowledge as he has become an experienced designer, so that he can make effective use of it in the professional design situations he faces. The disciplined approach taken towards design can be expressed in terms of a framework of possible activities, representing aspects of the design that the designer can attend to, thus giving a framework of intended actions within which to operate (Cross, 1989). In a model this may be represented in the form of a task structure. Design progression is critically appraised by the designer to detect unsatisfactory aspects and through these to provide information which will lead to improvements in the developing design, to the consideration of alternatives, and to progression towards completion of the design. This critical appraisal or evaluation of the design as it progresses is thus an integral part of the design process and plays a strategic role in determining what the designer pays attention to.

The structured technical domain knowledge and the task structure are closely related to one another and it has been argued extensively elsewhere (Keravnou, 1986) that these two aspects, viewed as components of competence in any professional field, must be represented explicitly in a knowledge based system if the system is to be capable of sensible dialogue and of providing meaningful explanations. (These are some of the capabilities which contribute to a multi-dimensional view of acceptable performance rather than the single-dimensional "right" answer view already mentioned in the context of first generation rule based systems). In the case of engineering designers, their task is further characterized by the constant critical appraisal of progression of the design to ensure that a satisfactory design will result efficiently. In assessing the adequacy of a knowledge based systems architecture for modelling engineering design tasks the extent to which each of these three characteristic components, namely technical domain knowledge, task structure and strategic use of critical appraisal, can be captured must be considered.

As design progresses, the designer acquires information which affects the way in which further progress is made or attempted. New information, on occasion, may change the status or confidence of decisions made prior to the new information's availability. The basis for a particular decision may be eroded all together, requiring the designer to

rework affected parts of the design (Asimow, 1962) or to rethink his design and consider alternatives. The competent designer is adaptable, as he learns more about the design problem he adapts his actions (Cross, 1989), he makes strategic choices about what to consider next. A knowledge based system modelling designers' competence needs to support this responsiveness to new information resulting from exploration of design possibilities.

In characterizing the task of engineering design, it is noted that discussions are often based on the notion of making minimum commitments and the idea that this is fundamental to successful design. The adoption of a minimum commitment principle is supposed to allow the requirements of a specification to be met whilst still giving room for manoeuvring among innovative, worthwhile solutions (Watts, 1966). Essentially, as a design progresses, according to this principle, commitments which unnecessarily constrain aspects of the design early on are not made. Akin's work on investigating the differences in approaches to the preliminary stages of design between designers and non-designers showed that designers actually explore a problem in depth before focusing on a solution. They consider to some depth the implications of even those ideas which do not seem a priori likely to succeed and they avoid adopting a solution until a number of strong alternatives have been considered (Akin, 1988). This approach maximises the freedom to explore alternative solutions to the design as a whole or to parts of it (Asimow, 1962). A further criterion for adequacy of the architecture of any knowledge based system which supports engineering design is that it should permit this way of working.

To be able to model design competence, a central question expressed by Mostow must be addressed, "How do designers decide what to do next? We need to uncover the reasoning behind such decisions and represent it explicitly" (Mostow, 1985, p.49). Most design problems are too complex for the designer to hold all the factors in his mind at once. The challenge which faces him is therefore where to begin and how to proceed, i.e. what strategy to use (Lawson, 1990, p.134).

> "Engineering design is essentially a matter of thinking of a number of alternative solutions to each problem. The designer's skill and experience is most vital at the points where he has to exercise his judgement in choosing the best alternative." (Rogers, 1983, p.65)

The designer needs to know what can be called into question (Lawson, op.cit.), he has to establish the scope of what he can design - what part of the world he has control over, his central activity is "understanding the field of the context and inventing a form to fit it", these concerns "are really two aspects of the same process" (Alexander, 1964, p.21).

A knowledge based systems architecture which supports competence modelling must acknowledge the central importance of representing both the reasoning behind design decisions and the knowledge structures which skilled designers have built up through experience to suit their purposes. This requirement is based on the premise that designers' grasp of the (strategic) choices available at any stage in design and their ability to know on what basis to choose exactly what to do constitutes their expertise. The expertise is characterized by the flexibility to respond effectively to the current problem situation (the context, the design, and their interdependencies) because what is relevant only becomes clear as design proceeds and depends on the decisions they take along the way.

In the remainder of this chapter some of the major problems with knowledge based systems for design (section 3.1) are firstly reviewed. This is followed by further examination of the roles and natures of three strongly interconnected aspects of engineering designers' competence. These concern how designers generate and evaluate design alternatives (section 3.2), how they make trade-offs (section 3.3) and how they justify design decisions (section 3.4). The interest in examining these aspects of designers' competence in detail is to explore the basic requirements for providing useful support for designers' activities using a knowledge based system based upon the Competent Systems architecture.

3.1 Problems with Knowledge Based Systems for Design

The problems discussed here should not be viewed as distinctly separate problems to be overcome by being picked off one by one. Some can be seen as different "symptoms" of a single "cause" and some interact more subtly with one another. They are dealt with under two separate headings below for the sake of clarity

3.1.1 Task characterization

Many knowledge based systems for design, particularly those which purport to do the designing, are influenced by first generation expert systems architectures and inherit some of their shortcomings (Clancey, 1983; Aikins, 1983; Keravnou, 1989) but in addition, poor task characterization compounds the limitations of the underlying technology. Knowledge based systems which have been developed for problems classified as design tasks are often based on modelling design as a well-structured problem solving task which can be represented (implicitly or explicitly) as a hierarchy of goals. The evidence against seeing design this way has already been given in chapter 2. Here that evidence is not repeated, instead the consequences in terms of limiting modelling potential are pursued.

From the perspective of representing design as a goal hierarchy "solving

the problem" consists of traversing the goal tree making use of procedures based on variations of the "establish and refine" model of task tree traversal (Aikins, op.cit.; Chandresekaran, 1983a). These procedures are applied in a recursive fashion to converge on a solution. It has been noted that a goal structure in the form of a strict hierarchy lacks the ability to adequately represent the interacting sub-goals which design tasks entail. Mostow (1985), for example, identifies several relationships among goals as ones which frequently occur in design activity and which are problematical from the goal-hierarchy model perspective. He lists them as goals conflicting i.e two goals cannot both be achieved; goals shared (one goal helps achieve another, other than its ancestor in the goal hierarchy); and goals having prerequisites i.e. one goal needs to be achieved before another in a different part of the goal hierarchy.

Where design knowledge is represented within a hierarchical goal structure, constraints between sub-goals are forced to be compiled into the design knowledge in a particular place in the goal hierarchy. The argument in favour of this is that suitable placement of constraints can lead to rapid focusing for problem solving. This has been described as a commitment to "putting the knowledge where it is used". This view is espoused, for example, by Chandrasekaran and his co-workers, and is exemplified in the architecture of AIR-CYL described in chapter 3. A further example of an application within this "school" is described by Brown and Breau (Brown, 1986) who applied this principle to the placement of constraints which they characterize as "implicit" constraints and which they represent by absorption, by compiling them into the design knowledge. This approach towards the handling of constraints, if applied indiscriminately, forces trade-offs to be artificially compiled into a goal hierarchy where they do not "naturally" belong and as with all compiled knowledge there can be problems with explanations and system extensions. (In fact the problem with explanations is explicitly acknowledged in later work by Kassalty and Brown (Kassalty, 1987). Because the task is partitioned into specialists, (a specialist, as defined in chapter 3 section 1, is a unit comprising factual knowledge combined with knowledge about how use it), the position of constraints is determined by the way the task has been partitioned - i.e. it is implicit in the decomposition of the task. This causes problems with the explanation facilities since as any explanation takes place within a context *determined* by the decomposition, the explanation facility is not capable of explaining the decomposition itself.)

It is widely conceded that trading-off between parameters or between component parts of a design is both difficult to handle in knowledge based systems and urgently in need of further research (e.g. Marcus, 1989; Monaghan, 1986). Because trade-offs are handled poorly in architectures which represent tasks as hierarchically structured goal trees, (see for example, the discussion of AIR-CYL in chapter 3 section 1.9.2.) they are frequently put to one side by announcing them to be outside the scope of the

work, or they are at best simplified so as to be accommodated. However, this has not proved to be a successful strategy since trade-offs seem to be intrinsic to design, they have been found to be inevitable even in design applications which have been particularly selected to be of the most "straight forward" type (Chandresekaran,1989; Marcus, 1989).

As a separate but related issue, decomposition of design tasks presents a serious challenge in its own right. Where decomposition is made in terms of goals (in many cases the actual term "goal" is avoided, the nature of the decomposition is not changed however by calling it something else), progress with the design is modelled in terms of establishing and refining parts of the design. The kinds of design processes that can be modelled like this are limited to convergent kinds. AIR-CYL is an archetypical example of this way of modelling design (see the discussion of AIR-CYL in section 1.9 of chapter 3). EDS, on the other hand, is unusual among knowledge based systems for design in its support for divergent exploration of mutually inconsistent design alternatives (see chapter 3 section 2.9.1).

3.1.2 Explanation and justification of decisions

It is well-established that the ability to provide adequate explanations is a desirable and essential characteristic of knowledge based systems if they are to be used successfully and gain acceptance among users (see for example, Murdoch, 1985). This requirement is well documented for domains where research on the application of knowledge based systems technology is most mature such as medicine (Hasling, 1984). Explanation potential, as pointed out earlier in this chapter, depends upon the degree of the structural relationships between the model represented in a knowledge based system and the components of expertise it models (section 1.4 above). In second generation expert systems specifically, this issue is seen in terms of the extent to which task structure and strategic choices are explicitly represented (sections 3.1 and 3.3 above). In knowledge based systems for design, lack of explicitly represented task structure, or poor characterization of it, limits the extent and quality of interaction which can be supported (see for example the discussions of EDS and Janus in chapter 3 sections 2.9.2 and 3.9.2 respectively).

Knowledge based systems for design make special demands concerning explanation and justification of decisions. In the case of designing there is a need to justify what is proposed by relating it to *other* design alternatives, that is to explain why one design has been proposed in preference to another design, or in preference to a similar design, and why trade-offs have been made in one way rather than another. These special demands arise because of the nature of the product - a design - which will be *one* effective solution rather than *the* sole, definitive solution to a problem or need.

Acknowledgement of this special status of a design proposal, as one possible effective solution, has as a natural consequence for knowledge based systems for design in that it renders them particulary open to criticism on the grounds of being limited in what they "know". Limited knowledge is a characteristic of all knowledge based systems, but in design where ill-structured problems are the norm, the knowledge limitation issue is a much more prominent one. The main limitations of models of knowledge cannot be overcome by adding more knowledge (a solution implicitly being pursued in the CYC project (Lenat, 1990) among others) or by adding "deeper" knowledge since, viewed as a model of practical reasoning, the reasoning in any knowledge based systems is inherently defeasible. We return to this issue in terms of the consequences for design support based on knowledge based systems technology in chapter 5, here it is sufficient to note that many designers of knowledge based design support systems deny this limitation and therefore do not pay attention to it in the design of their systems. As a result, critics often brand *all* knowledge based systems for design as not suitable for supporting designers (e.g. Fischer, 1991; Newton, 1988).

Having outlined some major problems with knowledge based support of design, the remainder of section 3 of this chapter gives further analysis of designers' abilities to make trade-offs among conflicting requirements, to generate and evaluate alternative design ideas, and to justify their design decisions and discusses the implications for knowledge based systems architectures capable of supporting each of them.

3.2 Making Trade-offs

Design, as has been observed, is characterized by the need to make trade-offs. Design requirements are always strongly interrelated and tend to be treated as such by a designer throughout design. Mayall (Mayall, 1979) has called this the principle of totality. Darke's studies of designers (already mentioned in chapter 2 section 3) bear out this idea. Designers see the various requirements as facets of a single design challenge. Alexander's seminal work on the synthesis of form (Alexander, 1964) is predicated on the understanding that a simple mapping between design requirements and features of a designed (solution) product is not routinely possible. One of numerous examples of findings reported for specific design domains which confirm this is the work of Tunnicliffe who has found that designers of page-layouts tend to think about legibility and readability requirements in a holistic way, as a consequence it is not a simple (or even meaningful) matter to associate these characteristics of a page with distinct features such as fount, weight, size, width, amount of text, type of paper, and so on (Tunnicliffe, 1990).

The trade-offs made by a designer are not arbitrary, they are design decisions which arise from application of design principles to a specific

situation. They involve professional and subjective value judgement directly (or indirectly via ranking) in terms of the importance ascribed to factors affecting a design (and indeed to the choice of what factors to attend to). The designer makes trade-offs on the basis of his experience, the purpose being to produce a satisfactory (or even good) design. Trade-offs may be ultimately traceable to generally applicable design commitments, they may be based on good practice applicable to a class of design activity. For example in mechanical design, there may be trade-offs based on commitments to minimise manufacturing costs, space utilization, material wastage, or overall weight (Pahl, 1988). Some trade-offs may be expressed as constraints which apply between components of the design. An experienced designer is one who has learned which constraints he should focus attention upon. He can make the constraints work for him by concentrating on the ones which will most constrain his design, thus he makes use of them to get an initial grasp on the design problem. Exploration based on ideas about the most significant constraints can be used to open up the major issues of a design situation. Thus constraints, whatever their function, can be used as generators of the form of a design. The way constraints are used, in which order, and with what emphasis, differentiates one designer (or perhaps one school of design) from another (Lawson, 1990). Lawson's observations of novice designers has shown that they may focus on the wrong kinds of constraints, wrong in the sense that they are minor ones, or well-understood ones, ones which do not help to structure the problem. Decisions about these constraints will be overruled by others whose effects influence the form of the solution rather than small details of it. (For an illustration of this see the example of the design of domestic architecture given in chapter 1.)

Many researchers have attempted to classify the kinds of constraints which are found in design activities. The basis of the classification varies according to the motivation for it. In individual knowledge based systems applications, classifications of constraints have been used to aid understanding of the possible roles constraints play in the design activity, and in these cases classification has been used to enrich the modelling activity (e.g. Brown, 1986; Frayman, 1987; Monaghan, 1986; Sriram, 1986a). However classifications derived for specific knowledge based systems applications tend not to have led to generalizations or abstractions. They remain application specific and cannot be readily applied elsewhere, nor have they necessarily given insight about more general application.

One classification scheme for constraints, developed by Lawson (op.cit.) to be used as an aid in understanding the nature of design problems, avoids many of the limitations (of lack of generality) which apply to those developed for one-off knowledge based systems applications for specific domains. In this general scheme, constraints are classified in terms of three dimensions: firstly, constraint generators such as legislators, users, clients, and the designer himself; secondly, functions of the constraints such as radical (those relating to the primary purposes of the designed

object), practical (those to do with production of the designed object), formal (to do with the form of the designed object) and symbolic; and thirdly, the domain i.e. whether the constraint is internal or external to the design. Using this classification scheme designs can be characterized in terms of the nature of the constraints which apply to them. A tightly constrained design is one where (for example) legislation and the client's requirements make heavy demands and there are many external factors affecting the design. An open-ended design is one where constraints are internal and tend to be generated by the designer himself.

Classification schemes motivated specifically for the development of knowledge based systems typically include types of local and global constraints and sometimes cater for trade-offs between parameters expressed as constraints. Local and global constraints are now considered in turn along with the way in which each is handled in knowledge based systems for design.

Local constraints include simple cases such as a parameter constrained within a range of values, or some straightforward relationships between parameters where once one parameter's value is settled the other's value may be simply calculated from it. These kinds of constraints can be accommodated fairly simply provided they can be represented by being placed suitably for consideration at a particular point or points in the design (decision making) task (even in one represented as a goal hierarchy). However, as has been noted already, even in design tasks chosen for their simplicity, problems have been reported when constraints have to be forced to be associated with the determination of one parameter (Marcus, 1989) or when they are compiled into a particular node of the goal structure (Kassalty, 1988). It appears that naturally occurring "local" constraints are rare, that is, ones which can be considered at a pre-determined stage of the design. Constraints are more often trade-offs involving two or more parameters interacting with each other, which have been forced by modelling restrictions to be localized. In some systems, to overcome recognised problems with trading-off, appeal is made to the designer-user of the system to resolve constraints which interfere with one another, perhaps by asking him to rank them in importance. For instance, Marcus and McDermott (Marcus, op.cit.) suggest that these constraints, which they term "antagonistic", should be detected as "unsolvable" (i.e. beyond resolution by a knowledge based design support system itself) and reported to the user for external resolution.

Global constraints apply to "parameters" such as cost, safety factors, and time. These parameters tend to be used to compare completed designs (and as such are considered to be in some sense external to the system producing the design). Alternatively, or additionally, global constraints may be represented implicitly, as ingrained assumptions and as such they are not available for explanation or justification of design decisions. Perhaps the most obvious example of this is the pervasive

influence of economy of cost which leads to descriptions of some designs as "better" than others when "cheaper" would be more accurate (Pye, 1964). Global constraints may be expressed as minimum standards to which an acceptable design must conform. If they are seen in this way, they are usually used simply to prune the set of alternative designs since a design will either conform or not conform to the minimum standard. However, in any realistic, interesting application global constraints can be traded-off against each other and against other design constraints. Once minimum acceptance criteria have been satisfied, factors such as safety, maintainability, reliability, cost, etc. can be seen as the vocabulary for describing the qualities of a design. This vocabulary plays a role in comparison of design alternatives and may be used in justification of design decisions and in making the case for favouring one design possibility over another. The relative importance accorded to qualities is a matter of professional judgement, open to discussion among professional designers, a place for normative arguments to be applied.

In non-trivial design activity, then, trade-offs cannot be avoided and that a designer will be engaged with decisions of this kind is inevitable. Any system which supports design must not obstruct this and ideally should support it. If it is the case that only the designer can decide trade-offs or prioritise constraints then a knowledge based system for assisting in design should at minimum be able to support him in making such decisions. This support can be provided, at least in part, by recording what decisions have been made, the circumstances (conditions) under which they were made and by telling the designer what other possibilities there have been and why they were not pursued. The Competent Systems architecture provides for explicit representation of the conditions under which strategies link the task structure into a graph dynamically during the design activity. It is therefore intrinsically capable of providing a strategic explanation facility. A trace of which (sub) tasks have been instantiated, which they were entered from, and under what conditions they were entered, can be used as the basis for a design log to record the design decisions made. Knowledge about what has been attempted (and why) supports designers in making decisions about what to attempt next (or what to rework or re-try under different conditions). Procedures which examine the design log and reason about the progress which has been made would support design decisions involving trade-offs and compromises provided that design commitments are explicitly represented and accessible for use both in the decision making and the recording of it.

3.3 Generating and Evaluating Alternatives

That a design idea is unsatisfactory is often, itself, useful information that helps the designer to know how to proceed with design. Indeed, Alexander (Alexander, 1964) observes that detecting misfits are the

primary data of experience and that the notion of "fit" - the suitability of a design - rests upon this. A misfit is a relationship between what has been designed (a design alternative) and what is known to be required. Designers can be seen to make a start on a design and make progress with it by working on those aspects which seem to demand attention most clearly (op.cit., p.26). Designers pursue ideas which they believe to be the important aspects of a design problem using them to structure the problem. They may develop a preliminary or outline design on this sort of basis and then examine the result to see what it reveals about the design problem itself (Lawson, op.cit., p.34). This process is what Schön (Schön, 1983) refers to as making use of the back-talk of the situation (see chapter 2 section 3 and the support for this given by Janus described in chapter 3 section 3.2). Schön argues that the reflection in action that characterizes much professional practice hinges on the element of surprise (op.cit., p.56). Having initially framed the problem in some way, the designer remains open to the discovery of phenomena which do not fit the initial problem setting. On the basis of the new insights he gets, he reframes the problem (op.cit., p.268), and explores it further. One source of problem restructuring is the detection of conflicts, another, identified by Akin (Akin, 1988) as particularly characteristic of designers, is prompted by the consideration of alternatives.

> "As the architect develops solutions or partial solutions that begin to meet some of the requirements of the initial problem description, comprehensive evaluations of these solutions are performed. Next the architect invariably alters the structure of the problem in ways which lead him to more successful results. A common form this restructuring takes is the addition or deletion of problem constraints or solution parts ... from the initial problem description" (Akin, 1988,p.179)[7].

It has been observed that designers are influenced by failure of decisions to fit the situation presented, decisions

> "are made as to whether or not a design solution satisfies the design objectives, if unsatisfactory, which part of the solution is at fault, what effect this fault has upon the solution, and the way in which such a fault should be corrected." (Derrington, 1987, p.22)

Brown and Breau (Brown, 1986) describe both the need to address the role of "routine failure" in the design process and the need to consider how such a failure triggers appropriate actions or provides information which can be used to proceed with the design. Focusing on constraint failures, they observe that it is usually the case that the manner of a constraint's failure suggests ways in which other values must be modified

---

7 cf. Gadamer, 1975, p. 236

to overcome that failure. In later, related work, Kwauk and Brown (Kwauk, 1987) describe extensions to DSPL, a knowledge representation language for routine design expert systems (described in chapter 3 section 1), to capture knowledge about how to handle constraint failures by explicitly representing failure suggestions and redesign knowledge at the points where specific failures can be corrected within their hierarchically organized structure of specialists. Chandresekaran and co-workers have identified that even in routine planning or design tasks knowledge about failure and how to recover from it is typically used by expert designers (Chandresekaran, 1989). A knowledge based system capable of supporting design needs to be able to accommodate and enable exploitation of knowledge gained through unsatisfactory attempts to advance a design.

In the Competent Systems architecture movement through the task structure is data driven thus there is the potential for making use of any new data, including that resulting from unsatisfactory excursions within the task structure, in furthering the design. Deciding what to do next during design is influenced by the current state of the design, by what has been tried so far, and by what has been learned from it. In a model of competence, viewed dynamically, progress with a design is determined by the current state of the design and the history of the (sub) tasks attempted (potentially supplied from the design log). Tasks are selected to execute strategies which themselves are selected opportunistically on the basis of enabling, disabling and relaxation conditions tested against the recorded progress of the design. Through explicitly represented strategies this architecture can support a model which captures Derrington's notion that "the information processing activity changes as design proceeds such that different strategies can be identified for the different phases of design" (Derrington, 1987). In the Competent Systems architecture, opportunistic strategy selection allows the determination of what to do next to be flexible in the sense of being directly responsive to the data (i.e. the current state of the design).

The ability to detect and to resolve unsatisfactory aspects of a design requires critical appraisal of what has been tried so far. The trace of (sub) task instantiations, the design log of decisions suitably associated with relevant design commitments, provides the raw material for this introspection. Procedures to trigger appropriate actions to resolve unsatisfactory design suggestions directly or to communicate with the designer about them based on the contents of the design log would support this aspect of designing.

Clearly, designers make use of other sources of experience outside the specific design task which is their current concern. For example, they may re-use or adapt case study material and they will take advantage of any previous experience with a similar design situation. A knowledge based system for design might call upon such resources, for example by making use of some kind of case-based reasoning (Hammond, 1989). These

resources for designing, ones where specific instances are made use of *directly*, are beyond the scope of the work presented here.

3.4 Justifying Design Decisions

Design carries with it the imputation of compromise. A good design is recognized as one which achieves the best practicable balance among conflicting requirements (Pye, 1964).

> "The requirements for design conflict and cannot be reconciled. All designs for devices are in some degree failures, either because they flout one or another of the requirements or because they are compromises, and compromises imply a degree of failure." (Pye, op.cit.,p.77)

Decisions have to be made on the basis of the (incomplete) information available. Good designs are based on design commitments which can be defended by comparison with alternative designs which can be shown to be a poorer compromise (in Alexander's terms, a worse fit), or to exhibit poorer or fewer qualities than the design proposed. Design commitments generally derive from a desire for economy, safety, reliability, the reduction of risk, and the obviation of failure (Petroski, 1985). In engineering design, the designer is always striving to overcome obstacles and problems. His task is a Sisyphean one (Florman, 1976). The designer makes choices based on the relative importance to be attached to each commitment and on the effect of each commitment on both the designed object and the design process. "One of the most important skills a designer must acquire is the ability to critically evaluate his own self-imposed constraints." (Lawson, 1990, p.71) This skill contributes to his ability to produce a design which he can justify to his professional peers.

For any design,

> "the requirements of use are imperative. If they are not complied with, the device does not give the result." But it is evident that some requirements are of a different kind for example, "the requirements of economy are on a different footing, for the amount of weight given to them is a matter of choice." (Pye, op.cit., p.35)

Clearly many possible designs or variants can be generated which fulfil the minimum acceptance criteria of utility (Lawson's radical functional constraints) but this is not where design as a skilled task ends. Fulfilling stated minimum acceptance criteria might be difficult or impossible but it could be described as "problem solving". It is the problem setting aspect of design which singles it out as a particular class of intelligent behaviour. Design criteria beyond functional usefulness are at

work in the design activity, and in the final choice of which design is selected and recognized to be a good one. A designer is always faced with some uncertainties and there is always more he could know about the design situation. The designer's task is to come up with a solution which is robust enough to withstand uncertainties and incomplete knowledge (Hogley, 1986). Many design failures can be traced to a misjudgment about what is important, what must be considered, and what can be ignored. Spectacular and famous examples of designs which failed for these reasons are the Tacoma Narrows Bridge collapse of 1940 and the Kansas Hyatt Regency Hotel walkway failure of 1981 (both described by Petroski (Petroski, 1985)).

In the final analysis it is the *design itself* not the process by which it was arrived at that matters, it is on the design's merits, and on the justification given for choosing it rather than choosing something else, that the design and the designer are judged. Eventually, the designer's decisions are presented for scrutiny, critics will not excuse mistakes or failures on the grounds of insufficient information or inadequate processes. Designers do not have the right to make misjudgments. The designer's choices are validated by his peers when they can recognise and accept the reasons for his proposals.

An exhaustive set of requirements can never be given, so justification of a design relies on two main factors. One is based on the relative merits of the design by comparison with practical alternatives, i.e. its *fit* to the situation in comparison with theirs. The distinguishable characteristics of a design are infinite, so for justification, relative misfits between alternatives play a crucial role. The second factor to be considered alongside the first is that among professional peers a great deal of unexpressed information is taken for granted. Justification of design decisions cannot therefore be taken at surface value, i.e. in terms only of what is explicitly stated. Professional norms and expectations determine what is expressed, i.e. what needs to be justified. For example, obviation of failure plays a central role in all designing. Design decisions, even those expressed positively in terms of the way they meet a requirement, can always be expanded with justification in terms of some failure avoidance properties which are offered by contrast with the properties of other design alternatives[8]. The role of justifications in supporting design practice is a matter which is developed further in chapter 5.

To support defence of a particular design, the design commitments brought into play, the ones salient for arriving at the compromise proposed, must be made plain. A defence may be expressed concretely by explaining

---

[8]Petroski (Petroski, 1989) makes a related point in the context of the role of failure in design. He proposes the avoidance of failure as a unifying principle on which the whole design process is predicated.

why one design is favoured over another. There is a crucial need for comparison of proposed designs (design decisions) with the alternatives. In a knowledge based system for design support design justification can be supported in part by a competence model which includes explicit representation of strategies and their logical bases. The design commitments which form the basis for compromises need to be clear. If these are represented explicitly in a knowledge based system, they can be used by it to reason about the design decisions which have been made and to support their justification. The "introspection" which this would afford would provide material to support defence of a design.

4 Conclusions

It has been suggested that a modelling approach can be taken to increase understanding of design behaviour and that to be successful in this the model must at least account for some aspects of design behaviour if it is to form the basis of a practical computer system (Coyne, 1990). The work presented here focuses on three strongly interconnected aspects of design behaviour, the making of trade-offs, the generation and evaluation of alternatives and the justification of design decisions. Any modelling to support these aspects of design activity appears to demand that rich conceptual structures be made explicit. These conceptual structures are supportable to an extent by a complex knowledge based systems architecture. Correspondingly rich computational structures are needed to represent a model of design competence on which to base a knowledge based system for design support.

Engineering designers make use of technical expertise relevant to their field, structured for effective application. This implies that the architecture chosen for a knowledge based system to support this process must be capable of representing technical domain knowledge suitably structured for the purpose of designing. Design activity is disciplined, the design process is organized to lead to the development of a design which can be defended in terms of its strengths in comparison with serious alternatives. A knowledge based system architecture suitable for modelling design activity must support description and manipulation of a representation of the task structure associated with design in the chosen domain and within which the strategic choices which are possible, and the bases for their selection, can be represented.

Competent designers are capable of critically appraising their design as it progresses. Their experience in working on the design is a resource which they use to help decide how best to progress. An effective knowledge based system for supporting design must be able to represent experience (information) gained as a design progresses and make use of it directly or make it available to support the design. This necessitates that

the system provide a means of using information about what has been attempted and about alternative choices as a contribution to critical appraisal, and to progressing the design.

There are strong links between generating alternatives and making trade-offs in design, and between the notion of critically appraising design alternatives as they are explored and defending the design finally proposed. The Competent Systems architecture offers a framework for exploring the support of these aspects of the designer's behaviour. It already provides the basis for a strategic explanation facility which gives concrete explanations in the form of a trace of the (sub) task instantiations which have occurred, and which uses the explicit task structure to give abstract explanations (ones related to purposes and plans). This facility in turn forms a basis for a design log in which to record the progress of the design, i.e. what has been decided, when, and why (and what were the alternatives). Procedures can examine a design log to inform decision making about trade-offs based on design commitments provided these are explicitly represented and associated with the design decisions in which they play a part or to which they relate in some other way.

To produce useful models of design competence it is necessary to study what designers do. Within the scope of the design situation with which they are presented they generate and evaluate alternatives and they make trade-offs to produce a good design, not committing themselves unnecessarily prematurely. They use domain knowledge, their past experience, and knowledge about what they have learned so far about a particular design problem, to produce a design. They are able to defend the results of their efforts by justifying the design decisions taken in the context of the alternative possibilities.

In the context of this description of designers' behaviour, empirical research has been conducted to improve understanding of how a designer behaves given real design problems in a particular professional setting and to provide a case study for illustrating the elements of a knowledge based systems architecture to model and actively support these facets of design competence. This empirical work is described in part 2 of this thesis. Firstly, however, part 1 concludes with chapter 5 which further explores the issues associated with modelling design competence and supporting designers' practice. Chapter 5 first addresses issues which lead to requirements for supporting design practice and then describes how the architectural components of a knowledge based system can begin to meet the requirements identified in this chapter and at the same time address the issues raised in chapter 5.

CHAPTER 5

Supporting Design Practice Using Knowledge Based Systems Technology

"Whose voice, no one's, there is no one, there's a voice without a mouth, and somewhere a kind of hearing, something compelled to hear, and somewhere a hand, it calls that a hand, it wants to make a hand, or if not a hand something somewhere that can leave a trace, of what is made, of what is said ..."

Samuel Beckett (Text for Nothing XIII)

This chapter discusses the support which can be offered to a designer by a knowledge based system based upon a model of competence. The emphasis is on exploring what is needed for successful partnership and co-operation between the designer and the knowledge based system which is *supporting* him in his work. To be effective, any support system must fit into the professional environment which sets the context within which a designer works. Professional practice both determines and is determined by what is important to a designer. Section 2 sets out the requirements for a knowledge based system for design support which follow as a consequence of a designer's professional accountability, his needs for reflective practice, and the prejudices which characterize a professional or a professional group. Section 3 focuses on the specific matters of how the making of trade-offs, the consideration of design alternatives, and the justification of design decisions can be supported by architectural components of a knowledge based system. Firstly, however, section 1 briefly states the relationship between competence and professional practice.

1 Competence in a Professional Setting

So far an attempt has been made to establish that knowledge based systems for design support can valuably be approached from the perspective of modelling competence. A knowledge based system based on a model of competence would include components which explicitly represent the task, the strategic choices which can be made to achieve parts of the task, and the bases for strategy selection.

The notion of competence in some profession cannot be separated from the organizational setting within which it is exercised. Professional choices are made for a purpose and take place in a context of professional practice conforming to, and suffused by, professional norms. Choices are not intelligible separated from this normative context. Explanations and

justifications given by professionals are imbued with professional norms. Likewise, the soundness of the advice given by a knowledge based system cannot be dissociated from its appropriate professional context. This implies that a knowledge based system should be designed in ways that make its place within the relevant normative context overt and plain to see. It should be designed with explicit structures which enable the user to situate both it and any advice it offers within a valid context. Explanation facilities have an important role to play here as they enable the users of a knowledge based system to assess the appropriateness of the advice offered. A model of competence should be based on a coherent interpretation of the nature of the task and make sense from the chosen professional viewpoint. The dialogue structure and explanations in a knowledge based system based on such a model can help the designer, as a user of the knowledge based system, to see the view of professional practice embodied in the model on which it is based. Explanations make known the reasons for a suggested action or conclusion by providing a trace of the argument. The designer-user, as a reasoning being, can relate to this form of interaction and thus can interpret the intentions captured in the model and make judgements about the justification of the reasoning presented.

An expression which fulfils the role of an explanation can only be understood by comprehending the function that it plays in a specific context. It should be clear from this, therefore, that explanation facilities cannot be added on to a knowledge based system as optional "embellishments" to the declarative and procedural knowledge representations from which it is constructed. Eliciting what constitutes an explanation in a context is an integral part of the knowledge itself, not something somehow separate, optional, or supplementary to it[1]. There are consequential implications for how knowledge elicitation should be conducted both in terms of what kind of questions should be asked and how the knowledge elicited must be represented to capture the functional and systemic qualities of the terms used. These matters are pursued further in chapter 6 in which knowledge elicitation methods are described and illustrated.

2 Requirements for Supporting Design Practice

In this section are considered the effects on knowledge based systems for design support of the fact that a designer practices his profession, by behaving in an intelligent, responsible and effective way, in a social setting. The following are considered in turn: that a designer, in practising

---

[1]It is important to grasp that a fundamental role of explanations is to remove or to avert misunderstanding (Wittgenstein, 1953, §87). In fact it is this aspect of an explanation which lets it stand on its own, so to speak, as empirically adequate without infinite regress via explanations of explanations, ad infinitum. Explanations are, therefore, inherently both strategic and contextual.

his profession, must account for and take responsibility for his decisions; that he will consciously reflect about what he does and must be able to do so if he is to be able to cope with situations he has not faced before; and finally that his work will be defined and framed in terms of the standards and norms of a professional setting.

2.1 Professionals are Accountable

At the most basic level, professionals are accountable for their actions on the basis of being humans, essentially social beings. As social beings, humans understand and interpret the behaviour of others on the assumption that they act as social agents.

> "Agency has to do with the mutual accountability of human conduct. This mutual accountability is one of the most significant features of organized social experience. Human beings describe and explain their own and others' activities for a host of practical purposes, and in the light of a host of relevant circumstances occurring in everyday life ... When we employ action concepts we are unavoidably engaged in ascription, imputations or appraisals." (Johnson, 1991)

Social reality cannot be transcended by agents acting in a social world, since they (we) are constituted by it. Consequently we are all committed to take up positions, to ascribing of responsibility, to attribute motives and to make potentially argumentative declarations. We cannot describe action other than socially committed action (Johnson, op.cit.). Human behaviour cannot therefore be described in some "context free" way which is categorically immune from revision.

Professionals who are competent to practice in their chosen sphere are agents in this unavoidable general sense to whom refinements of the general notion of agency apply. To be deemed competent in a profession prerequisites of acceptable training and experience are demanded. Evaluation of performance (actions, decisions) is measured against the professional standards of the appropriate qualifying body or recognised as acceptable by peers in the professional group. The normative influences of the professional group shape and affect the individual's behaviour. A model of professional competence will be set within the context of both the particular normative influences of the professional practice and the more general social context. A model which avoided social and normative commitments would not only be meaningless but is literally unimaginable. Consequently, the interaction enabled (in a computer system) on the basis of a model of competence reinforces the commitments being made in the (knowledge based) system which supports the interaction.

2.1.1 Professionals take responsibility

In "Mind Over Machine" (Dreyfus, 1986) the Dreyfus brothers identify five stages in skill acquisition spanning the range from novice to expert. They claim these as a common pattern that can be discerned in a wide range of disciplines and everyday activities in which individuals handle unstructured problems. (Unstructured problems, they define as ones which are not clearly defined in terms of goals, what information is relevant, and the effects of decisions.) The five stages can be briefly summarised as follows : novice - one who uses a restricted set of context-free rules to decide what to do; advanced beginner - one who begins to make use of experiences of rule application to refine rules to fit different situations; competent - one who consciously plans and organizes choices on a rational basis; proficient - one who uses intuition to "see what to do" but who deliberates over how to do it; and finally, expert - one who exhibits a fluid performance, demonstrating a mature and well-practised understanding of what to do in a situation. (Each of these stages is described in fuller detail in the endnote to this chapter.)

At about the competent level the Dreyfuses identify a change in relationship between the skilled individual and the problem or situation with which he is presented. Novices and advanced beginners use the rules they have learned, applying them once they have identified which ones are relevant to the given situation. There is a certain detachment, inherent in this style of behaviour. Novices and advanced beginners are not involved with the situation, they act, feeling little responsibility, according to the rules they have learned. The competent individual, on the other hand, has to make choices, to decide what is important, and thus in this behaviour the beginnings of a sense of involvement are present. Competent individuals feel a sense of responsibility. At the competent level, the individual makes conscious choices of what to attempt and how to attempt it, after reflecting on what are the alternatives (cf. chapter 4 section 1.3) and by this behaviour they retain some level of detachment.

The proficient performer has developed a perspective, "... certain features of the situation stand out as salient and others will recede into the background and be ignored. As events modify the salient features, plans, expectations, and even the relative salience of features will gradually change. No detached choice or deliberation occurs." (Dreyfus, op.cit., p.28). According to the Dreyfuses, proficiency and the first use of intuition results from serious involvement with a situation and from holistic discrimination in terms of "seeing the problem" whilst retaining deliberation in behaviour for solving it. Expert performance is characterised by "immersed involvement", at this stage the holistic view extends to become one holistic association covering both "seeing the problem" and "sensing the solution" . In summarising their view of skill as five separately distinguishable levels the Dreyfuses point out the following (emphasis added),

" What should stand out is the progression from the analytic behaviour of a detached subject, consciously decomposing his environment into recognisable elements, and following abstract rules, to *involved* skilled behaviour based on holistic paring of new situations with associated responses produced by successful experiences in similar situations" (op.cit., p.35).

Turning to design professions specifically, the same sense of immersion and responsibility can be observed once an observer becomes attuned to recognize it. Although a designer can be described as one who attempts to meet a situation not to master it (Potter, 1980), this "meeting" demands of the designer that he impose coherence on a design situation - making his own impression on it - and as a result "the designer must take responsibility for the order he imposes" (Schön,1983, p.163). Engagement with the design situation demands commitments to be made and these in their turn give the designer a sense of responsibility for *the way* he has engaged with the situation. Schön has described it thus - the practitioner as an enquirer is in the situation that he is trying to understand since it is partly, at least, of his own making (op.cit., p.150). The aspect of responsibility associated with immersed commitment is dealt with in the Dreyfuses' account of skill levels. That aspect associated with professional practice, the responsibilities associated with conformance to professional standards of conduct and the less formally defined but nevertheless influential norms of behaviour acceptable within the professional group do not emerge. (Reasons for this are suggested in the endnote.) Ability to give justification of actions taken and decisions made and communication with peers and others is not only demanded of a practising professional it is part of what constitutes one. Responsibility lies with the designer in this sense also. Although expert performance may to all appearances take place holistically, justification of the validity of what is done or decided may be called for. It is important to notice that the ability to rationalize in the sense of accounting for one's behaviour does not imply that the decisions made or the actions taken were *the result* of the same rationalization process[2] (also cf. endnote). It must be accepted, however, that a *formative* influence on the acquisition of professional expertise is the requirement to account for one's professional practice on occasion. The reflective aspect of professional behaviour which are explored in section 2.2 below provides the practitioner with much of the analytical skill needed to account for what he does. Before

---

2 These issues are often not separated. It is common to find insistence that to be able to give a rational explanation for some action necessarily implies that the act was the culmination of the same rational steps. Failure to make a distinction here can lead to simplistic knowledge elicitation methods. Professor Johnson proposes an analogy : the properties of a cake, once baked, render it suitable for cutting up into slices; yet no-one supposes that the cake has slices in it prior to baking. Applying the analogy to knowledge elicitation gives the observation that some methods elicit "slices" and the knowledge engineer has to try to make a "cake" out of them.

considering the reflective aspect of professional practice the nature of the facilities needed in a knowledge based system if due regard is to be paid to the responsibility which rests upon the professional individual are dealt with.

2.1.2 Facilities for handing back responsibility

An important aspect of the explanations given by a knowledge based system is the role they play in handing back responsibility to the professional user. The style of interaction should reinforce the idea that any practical reasoning is defeasible (chapter 4 section 1.5). The style of interaction supported by the Janus system (chapter 3 section 3) through its facilities for showing and allowing the user to add to an argumentation base is an example of one practical approach in this direction. Some knowledge based system designers working on classes of applications other than design support have also attempted to make their systems' limitations plain to users. A good example in the area of legal reasoning is the LEGOL/NORMA project (Stamper, 1987). The objective in this work was to assist lawyers flexibly, whilst accepting as fundamental that "the very essence of law's pronouncements is its scope for interpretation and negotiation ... tasks which are quintessentially human" (Althaus, 1989, p.315). Interaction in this system is designed to take place with the user in such a way as to acknowledges that "judgements and opinions belong to the realm of interpretive subjectivity and have to be exercised outside the structure of a computer" and by placing decision making firmly in the hands of the user also to ensure open-endedness at every level" (op.cit., p.324).

Explanations have an important function with regard to the validation of a knowledge based system's conclusions. In a particular situation this validation is assisted by helping the system's users to recognize the purposive aspects of the system's suggestions.

> "The purposive character of an action is fully recognised when the answer to the question 'what' is explained in terms of an answer to the question 'why'. I understand what you intended to do, if you are able to explain to me why you did such-and-such an action."
> (Ricoeur, 1981)

The forms of interaction have value in exposing the motivational basis for the suggestions. Explanations play an important part in assisting the user in evaluating the applicability, appropriateness, and adequacy of the design advice given. This characteristic is important because, as we have seen, responsibility for decisions will ultimately rest with the user (Boden, 1985; Dym, 1991).

A knowledge based system which has explanation capabilities clearly

cannot be in any sense an agent itself, however its interactions with the user can be devised to show the nature of the decisions taken, in abstract terms, and to expose the context in which they have been carried out in concrete, circumstance-specific terms (Suchman, 1987; Winograd, 1986). They play a part, therefore, in acknowledging the agency of the user.

2.2 Practice is reflective

The spontaneous behaviour of a skilled professional designer appears to be the result of "a kind of knowing which does not stem from a prior intellectual operation" (Schön, 1983, p.51). Dealt with at the more general level (common to all human behaviour) this observation is accounted for by Heidegger's[3] claims that practical understanding is more fundamental than detached, theoretical understanding. Essentially, the argument is that as humans our essential, primary experience of being comes from our being the world, before any sense of self distinguished from the world. Through our being in the world we apprehend what surrounds us and it is not open to us to choose to do otherwise.

2.2.1 Reflection supports learning from the unexpected

Fundamentally, then, we relate to things in the world by having them about us (ready-to-hand) and only when something unexpected occurs, when our attention is drawn, do things in the world become noticed as objects of our conscious attention (they become present-to-hand). Schön (as already introduced in chapter 2 section 3.3) describes professional practice as a reflective activity, in which the practitioner can move between intuitive and reflective responses as he comes to an understanding of a professional situation. The idea of reflection in action gives recognition to the observation that a professional can think (deliberate about) what he is doing. Much of the reflective aspect of understanding in a professional context is attributed to being a response to some surprising element in a situation. Reflection helps the professional to make sense of the situation in a new way (to reframe it). Reflection and action focus one another interactively (Schön, op.cit., p.56). As a result of this interaction, reflection provides a means of both shaping and exploring a situation (op.cit., p.269). The surprising consequences of action, reflected upon are what allows a professional to deal with new situations (new problems), to discover phenomena relevant and interesting, and to learn from his experiences.

> "When someone reflects-in-action ... he constructs a new theory of the unique case. His enquiry is not limited to a deliberation about means which depend on prior agreement about ends. He does not keep means and ends separate, but defines them interactively as he

---

[3]References to the work of Heidegger are based on the interpretations given by Kearney (Kearney, 1986), Steiner (Steiner, 1978), and the account given by Winograd and Flores (Winograd, 1986).

frames the problematic situation." (Schön, op.cit., p.68)

Kolb, in the context of understanding how humans learn from experience, handles the tension and interaction between tacit knowing on the one hand and the rational, critical and analytical aspects on the other by describing it in terms of a dialectic between what he calls knowing by apprehension and knowing by comprehension (Kolb, 1984). A cycle is observed in which one *apprehends* a situation (tacitly, here-and-now) as a situation recognised or defined by some (prior)*comprehension* of what it is relevant or important to attend to. Apprehension is improved from (later) comprehension, and leads to refinement of what is attended to :

> 'The enduring nature of the articulate forms of comprehensive knowledge makes it possible to analyze, criticize and rearrange these forms in different times and contexts. It is through such critical activity that the network of comprehensive knowledge is refined, elaborated, and integrated." (Kolb, op.cit., p.103).

Kolb's process of comprehension, like Schön's reflection-in-practice, is not a less important activity than tacit, spontaneous knowing, because it is the power of comprehension that leads to seeing things differently, that is which gives the ability of coping with new (problem) situations and of learning from them.

> "Comprehensions guide our choices of experiences and direct our attention to those aspects of apprehended experiences to be considered relevant. Comprehension is more than a secondary process of represented selected aspects of apprehended reality. The process of comprehension is capable of selecting and reshaping apprehended experience in ways that are more powerful and profound." (Kolb, op.cit., p.107).

It is important to grasp the ontological status of the phenomenon of being-in-the-world *underlying* the interpretations given by Schön and Kolb. Its description in terms of the reflection-in-action of professionals or in terms of a dialectic between different kinds of knowing through which we learn from experience does not reduce it to the status of a strategy which may or may not be used in practice. At the phenomenological level it is ontological. It is not therefore, an activity to be supported *optionally* by a knowledge based system. It describes what the system user *will be doing* regardless of whether or not the knowledge based system has been designed by someone consciously taking this into account.

2.2.2 Facilities to enhance designers' reflection

A knowledge based system is, in itself, finite and extremely limited through its unavoidable reliance on representation. What can be denoted by it is thus restricted and therefore the "world" which its representations

reflect is fixed and limited. The openness to respond effectively to something unforseen is therefore only a characteristic of the designer not the (knowledge based) support system. Winograd and Flores (Winograd, 1986) suggest that computer systems should be devised so that interactions promote connotation rather than denotation. In practice this would mean that the interaction should be oriented towards facilitating the evolution of understanding of the designer (as system user) rather than towards conveying system-made decisions about a situation which will inevitably be viewed by it in a very restricted.

> "A program is forever limited to working within the world determined by the programmer's explicit articulation of possible objects, properties and relations among them. It therefore embodies the blindness that goes with this articulation." (Winograd, op.cit., p.97)

The designer works with objects, properties and relations relevant to what he is doing, and these, when he deliberates about them, are present-to-hand. However what is present-to-hand at any moment shifts, as the designer's concerns shift, moving from pre-conscious experience of them as ready-to-hand (op.cit., p.97). The source of a computer system's blindness comes from its reliance on representation and prior articulation of what may be relevant.

There are two related consequences for the knowledge based systems designer. Firstly, what is represented must make sense to the user in both form and content, more precisely, it should be a representation

> "that deals with things the professional already knows how to work with, providing for precise and unambiguous description and manipulation. The critical issue is its correspondence to a domain that is ready-to-hand for those that will use it." (Winograd, op.cit., p.176)

Secondly, the interaction should, whilst making the limitations of the systems' knowledge clear, promote a sense of open-endedness, by being a resource to the *user* who is the party *responsible* for the judgements. The designers of the interactions supported in the LEGOL/NORMA system (already mentioned above in section 2.1.2) set themselves this objective. The argumentation aspects of the Janus system described in chapter section 3 can be seen as an attempt on the one hand to show the limitations of the system's knowledge whilst simultaneously "inviting" the user to make his *own* judgement using the advice available in the system as a *resource*.

2.3 Prejudices Characterize Practice

In the world of a practising professional designer there are accepted standards about what conduct is appropriate which include norms of

reasoning. A competent designer is one who is acknowledged to be so by those authorised to decide - his peer group or qualifying professional body for example. He is deemed competent on the basis of a judgement of his ability to achieve the standard of behaviour and to conform to the norms of the professional group to which he belongs. Professional designers routinely practice in branches of engineering where the tasks they face are complex and are not clearly defined. They operate in a changing environment to which their tasks must be adapted, and which is awash with value conflicts. This leads to differences in systems of appreciation *within* a community of professionals i.e. different views of professional practice. Because of the nature of what is being modelled, therefore, professional "prejudices" will be an *intrinsic* feature of any model of professional competence on which a knowledge based system may be based.

2.3.1 Prejudices are enabling

The design choices made by a designer at work are part of a process which is inherently dialectical. They are, inevitably, all about choosing and deciding in favour of some things and against alternatives (Gadamer, 1981). Choices are made on the basis of professional judgements competently executed. In chapter 2 it was suggested (section 4.1) that the design task can in fact be defined as exactly that with which the designer perceives himself faced. The design task is a construction which is derived from the *intentionality* with which the situation is impressed. It results from the designer's conformance to the beliefs and values of the professional group to which he belongs and within which he is considered competent to act (Merleau-Ponty, 1962).

Competent designers do not approach a new design brief from an arbitrary perspective or an uninformed viewpoint. The prejudices which the designer brings to a situation are an *essential* component of what enables him to see it the way he does (Gadamer, 1975) as a certain kind of task or challenge and to see it as a situation in which he is able to practice his skill, and to make professional judgements. It sets the framework within which he can make evaluations on the basis of some appreciative system (Vickers, 1965, 1968).

Thus a model of design competence, to be recognised as such, will embody the perspective provided by the enabling prejudices of the designer, the school of thought within which he or she operates, and the appreciative system of his professional group. However, the basis on which the model itself is valid, in particular, the basis on which the interaction it supports is predicated, is not contained within the competence model itself. In the actual professional environment which is being modelled the justification of the expertise is derived from the *background* of the professional and cultural context and the assumed, shared value system within which the expertise is exercised in the professional practice. (The limitations *inherent* in models of expertise have already been dealt with in chapter 4 section 1.4.)

2.3.2 Facilities for revealing prejudices

In a knowledge based system a trace of the argument, representing the choices made, in terms of the strategies considered but *dismissed* in specific circumstances *reveals* the competing possibilities represented within the model. References to the alternative possibilities can support the case for the reasoning that has been invoked. Dialogue in the form of a tracing of the argument exposes the boundaries of the model not solely by revealing what has been chosen, but from what alternatives represented in the model the choices have been made. The model itself, suited to a purpose, is derived from a perspective, in a context which should come from some coherent interpretation of the nature of the task - one which embodies the prejudices of the designer or group of designers from which its components were derived.

A competence model is, by definition, constructed from the competent professional's viewpoint. The reasoning about a situation captured in the model is intended to make sense, to be relevant, to someone interpreting a situation from the same professional viewpoint. The grasp of the relevant facts as represented in the model, the concepts and the terminology used, the context in which they are used, the priorities assigned, the differences which are distinguished, all serve to characterize the design situation in a particular way. The model makes sense within a particular interpretation, one that is useful for a particular kind of design practice and which is used by those engaged in it. The interaction supported by the knowledge based system has a key role to play in re-enforcing the characterization of the norms of reasoning, criteria of acceptance and grasp of content which constitute the competence model (Johnson, 1984). The dialogue structure of the knowledge based system helps the humans interacting with it to see the view of professional practice embodied in the model on which the knowledge based system is based.

Explanations play a role in assisting the user to inspect the limitations of a model by revealing the underlying assumptions on which its reasoning is based. It is through this means that the system meets the user's need to be able to decide when the model's conclusions are inapplicable due to external factors which can relevantly be brought to bear in a particular case to defeat the arguments represented by the reasoning in a model.

Explanation facilities reveal the reasons for suggested actions or conclusions and hence provide evidence of the coherence of a model within the modelling limitations. They help the user to decide whether the suggested design decisions are warranted in the situation to which a model has been applied. The explanations help the user to validate the knowledge based system's conclusions by calling upon the *user's capacity* to interpret the intentions captured in a model and to be clear about the model's limitations.

3 Components of a Knowledge Based System to Support the Making of Trade-Offs, the Consideration of Design Alternatives, and the Justification of Design Decisions

In this section some components of a knowledge based system that can play important roles in supporting a designer's need to make trade-offs, to consider alternative designs and to justify his design decisions are described. The competent systems components, that is the explicit task structure, the data driven reasoning and the static and dynamic modelling capabilities which are described in chapter 4 (section 2) are assumed as the basis for what is described here. These components, if implemented in a knowledge based system for design support, will be able to assist a designer by keeping track of decisions made, the circumstances giving rise to them, and the broader decision context (to the extent of showing the choices from which decisions were made).

Desirable design qualities, i.e. "good" design practice expressed as soft constraints, represented as *explicit design commitments* (described in section 3.2 below), with the mandatory requirements explicitly represented as hard constraints, will assist the designer in reflecting on his own professional judgement and in trading off design qualities accordingly. A *design log* , (described in section 3.3.2 below) enhancing the strategic explanation facility supported by the Competent Systems architecture (described in section 3.3.1 below), will form a resource for supporting a designer in trading off by presenting the relationship of the design so far with design qualities explicitly represented in the knowledge base.

A system with the elements described below will support the designer by relating specific design decisions to explicitly represented design commitments. Design commitments, brought to the designer's attention in a timely and relevant manner will support his decision making, and will be recorded as having played a part in determining the alternatives considered. By these means the designer will be assisted in informed trading-off between competing design considerations and in comparing the relative merits of alternatives. By recording (in the design log) what he has taken into account in decision making he will be supported in giving a rationale for a chosen design.

3.1 Explicit Task Structure

The Competent Systems task structure, which as a component of knowledge based systems architecture has already been described in chapter 4 section 2, provides the essential organising framework, the one that makes sense to a user, around which all the other components

described below are constructed. The evidence about how designers work indicates that the generation and evaluation of design alternatives are strongly interconnected and mutually supportive activities, a view not incompatible with the observation that from time to time they also consciously formally "stop and check" aspects of a design. The close iteration in Janus construction and argumentation (chapter 3 section 3.7) and evidence reported there that the designer needs to be alerted to information relevant to the design as it proceeds suggests that issues relevant to a design should be "triggered" i.e. drawn to the designer's attention as a design evolves, not in a separate evaluation cycle.

In a knowledge based system where there is explicit representation of task structure, there is potential to give guidance which is focused since there is the possibility of limiting the scope and direction of the advice offered. In chapter 3 section 3.9.2 the problems in the case of the Janus architecture of giving this sort of focused support were mentioned, the problem there is that the support system has no "knowledge" of the designer's task and therefore cannot give informed suggestions.

It has also been shown that a simplistic view of task structure (usually taking the form of a goal hierarchy) is too constraining for design tasks, the representation of task structure needs to capture what can be done but must not constrain the designer to work in an unnaturally inflexible manner. AIR-CYL's dynamic operation, of a hierarchy of non-interacting goal decompositions (see chapter 3 section 1.9.1) leads to only one possible design emerging from each input set of requirements, this is deemed acceptable on the basis of the "tight" specification of requirements (fully specified) and the fact that there are no designer choices. A task structure of the type supported in the Competent Systems architecture has an enabling role to play in the evaluation of alternatives through its support for representing alternative strategies and the basis for choosing between them. (We note however that the Competent Systems task structure does not support *simultaneous* exploration of different alternatives in the way that EDS does but as in the case of Janus it is a simple implementation matter to permit different alternatives to be considered and stored and retrieved for comparison and presentation according to the demands of the design environment.)

3.2 Explicitly Representing Design Commitments

In the view of designing which has been presented here it is the case that there is no simple mapping from the components of a design specification to the features of a completed design. On the other hand, the features of a design, and the particular trade-offs which are made are not arbitrary. They arise from the physical constraints which hold between components and reflect design commitments (design qualities that are valued) applied to specific situations. It has also been noted that constraints, from whatever source, are not only a negative force, ruling out

design possibilities, but that they are put to work "positively" by the designer as a means of structuring the design problem and rendering it both manageable by him and comprehensible to him.

Design constraints, in the widest sense, can be divided into two broad classes. On the one hand are those which set minimum acceptance criteria for a design i.e. criteria which can be used to rule out or to qualify a design for further consideration, and on the other hand, there are those which relate to qualities of the design. The former are "hard constraints", ones which usually must be satisfied "at all costs", that is to *qualify* a design for consideration at all. Legislation, safety requirements, and acceptable tolerances are examples of sources of these kinds of constraints. On the other hand, "soft constraints" are used to *compare* design alternatives and to *evaluate* how good a design is. The relative importance of these soft constraints, the qualities a design embodies and the extent to which it embodies them, is a matter of subjective judgement to which normative arguments apply. These arguments are generated and resolved by the professional group or individual designer and have their source in professional norms and values.

Defined in these terms, hard and soft constraints should be differentiated from one another clearly so that, among other things, they may show the designer where scope for trading-off lies and what is its nature. The choice of design qualities upon which to place most emphasis is the judgement of the designer; he can be helped to decide by seeing the choices available. The support for argumentation given by systems with the Janus architecture (chapter 3 section 3.6) is one example of an approach towards supporting the designer in choosing priorities among design qualities. Reminding the designer that he is making a judgement on the basis of his preferences by presenting them to him will help him decide *consciously*, aiding his *reflection* about what he considers important and why.

Let it be clear that the concern here is not with simple (non-contentious) local constraints among parameters (e.g. the relationship which holds between weight, volume and density). Something like the support offered by EDS (chapter 3 section 2.6) would be satisfactory for detecting, and drawing to the attention of the designer, these sorts of objective inconsistencies. The assistance which EDS offers to overcome detected inconsistencies is similar in style to part of what is supplied by the strategic explanation facility described below in section 3.3.1 (since EDS gives a trace of the inferences leading to the establishment of a parameter when requested to do so by a designer). The interest is more concerned with supporting the subjective or normative judgements of the designer or design group. These are the judgements which play an important role in choosing between design alternatives on the basis of the qualities rather than the essential functionality of a design.

Lawson's classification scheme for constraints, described in chapter 4 section 3.2, is intended to be an analytical aid to the understanding of the nature of a design problem and to provide a vocabulary for differentiating the spectrum of design problems from, at one extreme, tightly constrained ones, to, at the other, open-ended, under-constrained ones. Lawson's scheme can be appropriated to provide a classification of design constraints by shifting the focus to an analysis of the source of, and nature of the qualitative design commitments which apply to a particular design situation. Explicit representation of design commitments in a knowledge based system might be used to assist a designer in reflecting upon and choosing between conflicting commitments. Information about each constraint could include a classification in terms of Lawson's three dimensions namely, the generator, the function and the domain of each constraint. Information, in the form of a qualitative ranking of constraints could be provided from this sort of description (if applicable) and if the ranking were brought to the designer's attention at the relevant point in designing it might assist him in focusing his attention on the most firmly constrained aspect, helping him to chose from alternatives and to see the implications of his choice, but still leaving him free (through the form of the system's interaction) to make subjective judgements about the relative importance, in a particular design situation, of the qualitative constraints applicable to a design problem.

A designer has to make choices based on the relative importance he ascribes to the various, inevitably conflicting, design commitments. He can critically evaluate his own judgement about this if he is a good designer. His design choices are validated by peers when they can recognize and accept the reasons he gives for the proposals he makes. Justification relies on a comparison of the proposals with their practical alternatives on the basis of their relative fit to the design situation. Relative fit is expressed in terms of design qualities as well as the degree to which essential functional requirements are met. Implicit in the description of relative fit, i.e. in the making of a case for a particular proposal, is the relative importance attached to design commitments; these are both set and judged in terms of the professional norms and expectations of the professional group. The vocabulary used to make a case for a design rests on a shared commitment to what is important and what can be taken for granted.

A knowledge based system which has a representation of the task structure can use the knowledge the task structure represents to make sensibly focused associations between decisions being made, design qualities affected by or which affect them, and other information contextually relevant to the decision currently being made. The task structure therefore provides the crucial organising framework through which access to other key components of the design support system are provided in an intelligible way.

To summarize then, links established between design commitments and

the decisions to which they relate can support design in three interconnected ways. Firstly, drawn to the attention of the designer, at an appropriate point in his working, they can inform his decision making and encourage reflection on judgement of their relative value and importance. Secondly, they present a vocabulary for evaluating and comparing design alternatives and thirdly, if recorded alongside the decisions to which they relate they provide material for justifying design decisions.

In chapter 8 of this thesis it is demonstrated how design commitments and other information relevant to design decisions can be represented (as object hierarchies) and that these representations, set in the context of a representation of what the designer is trying to do, can be linked effectively (by message passing between object hierarchies) to support the designer's decision making.

3.3 Explanation Facilities

The dialogue structure (the user interface including the explanation facilities) supported by a knowledge based system is clearly restricted by what is explicitly represented in it and its form is determined by what is represented (Keravnou, 1986,1989). It has been established that good explanation facilities include strategic explanations about why a line of reasoning is pursued at a certain point in the problem solving activity or how a part of the task was concluded. Strategic explanations which are based on a model of competence can explain to the user the reason for conclusions in a form to which the user can relate naturally (chapter 4 section 2.3).

Explanations provide reasons for a conclusion, suggested action or state of affairs (Southwick, 1991), but these alone do not justify the reasoning although they have a role to play in a justification. Explanations will play a major part in allowing a designer as a user to see the scope of a knowledge based system which is supporting his work, the purposes for which it is suitable, and the situations in which its reasoning is valid. In short, explanations help a user to assess the appropriateness of the advice a knowledge based system offers (Swartout, 1989).

A knowledge based system in which the design task structure is represented explicitly offers the potential for explanations which can relate each step taken to the context of the design activity as a whole. These explanations can have two aspects the first of which makes reference to the particular practical circumstances which apply to the design activity currently being supported. This is the concrete aspect of the explanations. The second aspect reveals what is (theoretically) possible in terms of the avenues which could be explored, the alternative choices available, i.e. what could be considered to further the design. This is the abstract aspect of the explanations. A trace of the steps taken to arrive at design decisions will have these two aspects, the particular path taken and the abstract

structure of what was possible at each decision point.

Explanation facilities can be used to provide evidence about the validity of a suggested idea, recommendation, or conclusion by making reference to what is known about the particular design situation and by making reference to the alternatives that have been considered along the way. The explanation facilities trace the argument, including the premises on which it is based, which is the result of the model's reasoning. By this means, the explanation facilities provide for the user to see the reasons for belief in the system's conclusions and for valuing its recommendations.

> "The reason why a valid argument is a good reason to believe the conclusion is because the premises of a valid argument are sufficient for the truth of the conclusion." (Johnson, 1984)

It is through this link that the abstractions in a model and the psychological state of a user are related. The premises in the system are, within its modelling limitations, sufficient for its conclusions, and thus a user inspecting the explanations and finding nothing untoward, is entitled to believe the conclusions. Knowledge based system users are satisfied by justifications provided in this form because this is what they can respond to as reasoning beings. They are justified in believing reasoned facts explained to them by a knowledge based system if they do not have other over-riding information or evidence that the facts are not true.

On the basis of these premises, the sorts of explanations supported by the strategic explanation facility of the Competent Systems architecture are now described (in section 3.3.1 below). This description is followed by a discussion of the nature and role of the design log in a knowledge based system for design support (section 3.3.2).

3.3.1 Explaining "why", "how", and "why not"

The potential to provide answers to queries of the form "why?", "how?" and "why not?" of a knowledge based system which has explicit representation of the design task structure and the strategies which can be used to achieve parts of the design has already been introduced in chapter 4 section 2. Why-type explanation are concerned with explaining why a particular line of reasoning (designing) was pursued. An explanation can be given which includes abstract components provided from the explicit task structure and concrete components which explain the particular circumstances (the design data) giving rise to the specific approach attempted. Explanations of this kind make reference to the available choices, the basis for choosing among them (the abstract terms), and the particular circumstances that satisfied the conditions for the strategic choices which have been made (the concrete terms). Using these resources, explanations about "why" particular design decisions were made can be provided in the context of the options that were available.

How-type explanations use the same resources to describe how a particular conclusion about the design was accomplished. These explanations make similar reference to the choices available and how the specific circumstances which applied to a particular situation determined the way reasoning proceeded. In the context of a particular why-type explanation the reasons why alternatives were not pursued can be afforded through a why-not-type explanation which describes, in abstract terms, why particular alternative considerations or design strategies were not selected when they appeared as options at certain points in the reasoning. This last kind of explanation can only be provided by a knowledge based system in which alternative choices are explicitly represented, since to explain why a choice was not made the system must be able to refer to what alternative options were available at a decision point as well as to the one which was selected in the particular circumstances.

3.3.2 Design log

Whilst the abstract, strategic aspects of the explanations are supported by the task structure, the concrete, situation-specific aspects come from recording the information giving rise to the decisions made in supporting a particular case of designing. A trace of the parts of the task structure visited, each associated with the contextual information giving rise to the attempt at the sub-task and resulting from it, will form the core of a design log capable of providing the explanations of the types described above.

If the design log is able to record the links associating design commitments to the design decisions which were informed by them and/or which lend support to them this aspect of the design log would be a useful resource for justifying a design, and for weighing up the relative merits of alternative design decisions and by extension for comparing one design with another. Provided that access to the contents of the design log is made available during design in a flexible manner - i.e. under the control of the designer - it can be used as a resource by the designer in reviewing the design decisions he has made. To support his reflection, it would need to record the contextually relevant information associated with a decision, not only the design commitments which apply but also the aspects of the design situation more broadly which has been taken into account e.g. the requirements (either pre-specified or designer generated) which it satisfies.

A design log could provide temporal continuity (an implementation matter) by yielding a record of design decisions made during a session with the knowledge based design support system. By extension it could therefore form the basis for automated production of summaries and rationales for a number of competing designs for the same design situation.

If the design log is to be a flexible resource at the disposal of the designer

to support his work it is more appropriate to see it as a set of procedures than a (static) trace of each step taken. A trace in some form would of-course be needed to satisfy some of the design log's functions. There is a parallel here with the strategic explanation facility originally proposed for systems with the Competent Systems architecture. The components of the model, i.e. the task structure and strategies and the representation of the bases for choices provide the "strategic" or "abstract" aspects of the explanations whilst the trace (in the Competent Systems architecture, the trace of (sub-)task instantiations) of what has been done in a particular case gives the "concrete" or case-specific data for "instantiating" the explanations given. In the case of a design support system there will be an abstract representation of the design situation including task structure, design requirements, other constraints of various kinds - "soft" and "hard" and any other relevant contextual representable aspects. This forms the "strategic" resource for the design log procedures. The trace(s) of design decisions taken in a particular situation together with data which characterises that situation will give the situation-specific material to be used by the design log procedures to particularise the support it provides the designer.

In chapter 8 of this thesis it is demonstrated how a design log can be implemented in an object-oriented knowledge engineering environment. Although the design log implemented is rudimentary it does demonstrate how it constitutes a component of the architecture which functions at both an abstract and concrete level. The implementation also demonstrates the value of viewing the design log as a resource for delivering services to a designer to be used by him flexibly to support design in different ways.

4 Review

This chapter has considered the support which can be offered to a designer by a knowledge based system based upon a model of competence. A number of complementary aspects of the professional environment which sets the context within which a designer works have been explored to establish the consequences for the design of any knowledge based system useful for supporting his practice. Components of a knowledge based system based on a competence model which would support the interrelated activities of making of trade-offs, considering design alternatives, and justifying design decisions have been described. Examples of how some of these components can be implemented in a knowledge engineering environment based on the object-oriented paradigm are demonstrated (for the design situation of the case study of part 2) in chapter 8.

It is important not to shy away from studying real design activity in the natural setting of design practice. To even begin to do something useful towards supporting designers using knowledge based systems technology the first task is to find out what are the best ways of eliciting what design

tasks entail. An understanding of the design task is a pre-requisite for modelling competence and thus for building knowledge based systems to support design based upon a competence model. In Schön's terms, it is necessary to go down into the swampy lowlands to deliberately become involved in the "messy but crucially important problems" of the designers whose concerns are themselves with swampy lowlands (Schön, 1983, p. 43). The outcome of the research may not be "neat and tidy" - for example, a piece of software that, given some input, designs some micro-world artefact by way of an output - but it can be hoped that the outcome will be to move further forward the understanding of where knowledge based systems technology can *realistically* make a contribution in the *real* world of the practising designer.

Part 2 of this thesis comprises a case study which is a medium for exploring and illustrating the ideas and proposals presented in Part 1. Part 2 (consisting of chapters 6, 7 and 8) demonstrates how aspects of designers' behaviour can be investigated using appropriate knowledge acquisition techniques and how the proposals presented in Part 1 can be implemented using a typical state-of-the-art knowledge engineering environment. Chapter 9 briefly summarises the work presented in parts 1 and 2 and presents the conclusions.

End note to chapter 5 :

Further description of the Dreyfuses' account of five stages of skill acquisition and comments on it

In section 2.2.1 of this chapter the five levels of skill acquisition described by Dreyfus and Dreyfus (Dreyfus, 1986) were summarised as :

novice -	one who uses a restricted set of context-free rules to decide what to do;
advanced beginner -	one who begins to make use of experiences of rule application to refine rules to fit different situations;
competent -	one who consciously plans and organizes choices on a rational basis;
proficient -	one who uses intuition to "see what to do" but who deliberates over how to do it; and finally,
expert -	one who exhibits a fluid performance, demonstrating a mature and well-practised understanding of what to do in a situation.

These are now described and discussed in more detail.

The novice "processes" facts and features of a situation, ones pre-specified to be relevant, by objectively applying rules which specify what to do on the basis of those facts and features. Application of the rules to situations allow the novice to begin to accumulate experience. Practical experience in applying the rules to different situations leads towards the next stage, that of becoming an advanced beginner. An advanced beginner learns from experiences and refines the context-free rules used by the novice by adding conditional considerations which begin to take account of different situational contexts.

The Dreyfuses do not hold that an advanced beginner can necessarily verbally articulate the refinements to their decision making which result from the build up of concrete experiences. However, they have found that the number of context-free rules combined with situational considerations based on experience eventually become overwhelming for the advanced beginner. A sense of what is important in a given situation becomes necessary as a means of organising what is known. The knowledge needs to be structured for use. At this stage a sense of planning and of developing a strategy for approaching situations characterises the transition to the third skill level, that of competence.

> "Choosing a plan is no simple matter for the competent individual. There is no objective procedure like the novice's context-free feature recognition. And while the advanced beginner can get along without

> recognising and using a particular situational element until a sufficient number of examples renders identification easy and sure, to perform at the competent level *requires* choosing an organizing plan. Furthermore, the choice crucially affects behaviour in a way that one particular situational element rarely does." (Dreyfus, 1986, p.26)

A model of how an individual operates at a competent level to choose strategies on the basis of deciding what information (or what pattern of data) is important from a situation should therefore be representable in a suitably sophisticatedly, suitably structured information processing model.

At the proficient level of performance, the Dreyfuses introduce the notion of the "performer's perspective". Comparison should be made here with the notion of enabling prejudices (discussed in particular in section 2.3.1) and with the findings of researchers investigating designers' behaviour (see the findings of Akin, Rowe, and Darke in chapter 2 section 3). The proficient individual, then, responds intuitively to patterns without decomposing them into component features, the Dreyfuses refer to this as "holistic discrimination and association". The proficient individual's behaviour is still open to explanation in terms of facts and inferences. The proficient operator, as they define him or her, recognises a problem intuitively but consciously deliberates about it, deciding what to do.

Expert performance is characterised by "immersed involvement" and is routinely non-deliberative in nature. However, "... when time permits and the outcomes are crucial, an expert will deliberate before acting. But ... this deliberation does not require calculative problem solving, but rather involves critically reflecting on one's intuitions" (op.cit. p.31-32). A proficient performer who gets sufficiently varied experience appears to advance from a holistic approach towards "seeing the problem" to a more comprehensive holistic view which embraces not only "seeing the problem" but also "sensing the solution" or approach to a solution in one holistic association. Their view of expert performance is that, "elements and principles play no role in mature, practised decision making". They conclude from this that rationalization in this sense amounts to the *invention* of reasons, so here they are using the term rationalization in a derogatory sense - as rationalistic in a reductionist sense - rather than in the sense of giving a rational explanation.

The Dreyfuses' characterization of skill into five levels is intended to encompass skills in the most general sense of the term - they include every day skills like driving a car as well as professional skills like diagnosing diseases. An integral part of professional skill is communication and, when needed, justification, of actions taken and decisions made. The requirement to communicate with peers and others is, itself, part of the expertise, proficiency or competence in a most professional domains. In the professions then, (recalling chapter 4 section 1.2), "exercise" of expertise

cannot be sensibly separated from its "communication" (chapter 4 section 1.2) and therefore plays a part in determining its very nature. The Dreyfuses do not separately address this point. There is one further relevant observation worth pointing out here, namely that the phenomenon of using reflection in action to assimilate and make use of "surprising" or new experiences is not accounted for in the Dreyfuses description. Nevertheless, if their view that "competent performance is rational; proficiency is transitional; experts act arationally" (op.cit. p.36) is broadly accepted then the boundaries between what can and cannot be modelled in a computer system are set. The limits for representation lie at the level of competence models and this gives a guide as to when attention must move from *representation* of a skill to the *support* of a human who is practising it.

CHAPTER 6

Case Study Environment and Knowledge Elicitation

"'Look here , Taffy.' And he drew this - A. 'Now I'll copy it,' said Taffy. 'Will you understand this when you see it ?' And she drew this - A. 'Perfectly,' said her Daddy. 'And I'll be quite as s'prised when I see it anywhere, as if you had jumped out from behind a tree and said "Ah!"' 'Now make another noise,' said Taffy, very proud. 'Yah!' said her Daddy, very loud. 'H'm,' said Taffy. 'That's a mixy noise.'"

Rudyard Kipling (How the Alphabet was Made)

This chapter introduces the case study by describing the business and professional context within which the design work studied takes place. The nature of the design activity and the background of the designer are described. Aspects of the researcher's background which are relevant to the case study are also described. Section 2 gives an account of the series of interviews which took place, these were the primary source of knowledge elicited. A brief description of the documents which were also used as a secondary source of raw materials is given. Section 3 describes the knowledge elicitation process in some detail by describing the knowledge elicitation aids used and their roles in gathering, checking and analyzing the interview material. The chapter concludes by explaining the methodological basis for the knowledge elicitation approach taken.

1 Case Study Environment

The environment in which the case study took place is described in this section. The professional context within which the designer makes a contribution to the business overall is set out. A characterization of the designer interviewed and an outline of the nature of the design work he undertakes is given. The specific designs which were discussed during the interviews and for which documentation was made available are introduced. Finally, aspects of the researcher's background which affected the case study are reported.

1.1 Business Context

The study was conducted in a Regional Electricity Company (REC). Twelve such companies operate in the United Kingdom. Their main business is to supply electricity and to distribute it. This activity is regulated according to the terms of licences granted by the Electricity Regulator. The particular company in which the case study was conducted supplies

electricity to two million customers almost 80% of which are domestic premises. In a typical year it supplies a total of 19,000 GWh (gigawatt hours) of electricity.

The electricity distribution network is a major asset of any REC. Most of the electricity supplied to customers initially enters the distribution network at bulk supply points where it is purchased from electricity generating companies. From here it is distributed on what is called the primary distribution network (typically operating at 132, 66 or 33 kV (kilovolts)) to the REC's primary substations. At the primary substations transformers convert the electricity (typically) down to 11 or 6.6 kV for further distribution on the secondary distribution network to secondary substations where, for most customers, further transformation to low voltage (415 volts i.e. the 240 volts single phase supply usually provided to domestic premises) takes place. Final delivery to each home takes place via a low voltage distribution network.

1.2 Professional Context

The part of the REC's business with which the case study is concerned is the planning and designing of the primary distribution network. This network consists of the circuits which operate at 132, 66 and 33 kV and the primary substations which transform electricity from this voltage to a lower one. Substations essentially contain transformers and switches (switchgear) for operating the electrical network. However, primary substations are large and complex installations since they also contain equipment for the control, monitoring and protection of the substation equipment and the circuits connected to them. Subsidiary equipment such as voltage regulators and a variety of auxiliary plant e.g. for cooling, lighting, and providing emergency power supplies is also installed in these substations.

The requirements to supply and distribute electricity are constantly changing. Customers' use of electricity changes with time. Load demands increase or decrease in different geographical areas as building and land use changes. Requirements to re-route or divert cables constantly arise, for example as a result of road reconstruction and building development. Some part or other of the distribution network is always in need of replacement or upgrading due to deterioration from age, use, or the development of faults. These sorts of occurrences give rise to work for the engineers concerned with planning and designing the primary network.

1.2.1 Organization

Within the REC, the Engineering Director has overall responsibility for the construction, operation, maintenance, control, and planning of the whole electrical network. Separate departments handle each aspect of engineering with regular formal and informal interaction taking place between them. Informal channels of communication are important and

frequently used. Many of the engineering staff work in the same building and can easily contact one another without undue ceremony.

Within planning and design individual responsibilities are clearly established. There is clear demarcation of jobs and responsibilities, so that, for example, planning and design of parts of the network at different voltages are distinguished from each other and the design of substation layout is separately distinguished from other design tasks. Work is also divided by geographical area, so that a primary system planner will normally work on proposals for a particular area of the REC. For the case study the work known as primary system planning was selected.

When the need for a major extension or alteration of the primary network arises, it is the job of a primary system planner to investigate what is needed and to produce an outline design to meet the requirements identified. He will make use of a wide variety of sources of information including consultation with specialists inside the organization when necessary to arrive at a design proposal and to satisfy himself as to its suitability. However the process of arriving at a proposal - the design task - is essentially the work and responsibility of the designer alone. The outcome of his work on a particular matter is presented as a planning report which consists of a proposed design and evidence to support its acceptance. The designer's work in arriving at a proposal and in justifying it in a planning report is the focus of this case study.

1.2.2 The designer

Normally an engineer concerned with planning and design of primary networks will be very experienced in working on the network (i.e. as an operational engineer). It is common for a senior planning engineer to have perhaps twenty or more years of experience of working as an engineer in the same REC. He will have worked at sometime as a construction engineer managing and supervising new installations on the network and he will have worked as an operational engineer performing both routine switching operations on the network (e.g. to de-energise and isolate equipment for maintenance) and emergency switching operations to reconfigure the network when faults occur. Since it has been common until now for electricity supply engineers to remain with the same REC (or more accurately the same Area Electricity Board as they were formerly known) for much of their working lives, it is also common to find that senior engineers will have a very rich and detailed knowledge of the specific geography and physical composition of the networks with which their work is concerned.

The designer with whom the interviews for this case study were conducted fits this description. In addition to being clearly acknowledged to be an expert in his field on the judgement of his peers, he is considered, by them, to be particularly capable of devising innovative solutions when these are demanded. In practice, this means that "he's the one that gets given the really difficult problems" to tackle.

1.3 The Design Task

A problem requiring a designed solution is initially presented to the designer. The outcome of whatever proposal he finally arrives at must satisfy, i.e. overcome, the problem which was the initial spur to design work. However, the initial problem given may be poorly and incompletely described and more importantly the form of the design solution which will solve it cannot be simply, i.e. directly, determined from it. The designer's proposal must certainly "solve" the problem presented, but part of the design task is study the problem's *wider context* and to decide the *form* of the solution. The designer, therefore, is concerned with both problem setting and problem solving. Although it is an engineering design task in terms of the domain within which it falls, the nature of the task has many elements in common with preliminary design in other disciplines like architecture. Much of the behaviour which characterises the preliminary design described by Akin (Akin, 1988, cf. chapter 2 section 3) equally applies here viz. the examination of the problem in breadth before becoming committed to a solution; the debate of full implications of diverse ideas, even those which at first sight do not appear promising; and the avoidance of adopting a particular type of solution until a number of alternative have been considered. The designer studied also appeared to make use of scenarios of the kind defined by Akin - and coincidentally actually referred to these solution generators by using the term "scenario". The design task studied is very much concerned with the earliest stages of design, that is with problem formation.

Lawson (Lawson, 1990) has noted that in the design of mass produced objects the costs of design are significant in relation to the costs of the designed object. However, for one-off designs (he cites the design of individual buildings) the design process constitutes a very small proportion of the budget for producing the finished artefact. The conclusion he draws from this is that in the latter situation, there is no tradition of investing in efficient methods for design (Lawson, op.cit., p.205). Parallels can justifiably be drawn here with the one-off design of primary distribution networks.

1.3.1 Suitability for study

The nature of the task selected for study makes it a potentially rich source for investigating the aspects of design which are of most interest for this thesis. Firstly, the importance of not falling prey to the criticism of Dreyfus (Dreyfus, 1981; Dreyfus, 1986) concerning the limitations of studying artificial "micro-worlds" is taken for granted. It is assumed that it is wished to avoid "clever special solutions, which work because the real problems have been put aside" (Dreyfus, 1986, p.70). One aspect of the design task chosen renders it particularly suitable for study here. This aspect is a consequence of the nature of what is designed: the primary distribution network is a fundamental component of the electricity supply

system, it has to work efficiently and reliably. A newly designed part of it, once realised (installed), cannot easily be altered, recovered or removed. The new component must make the contribution to the whole in the way predicted and minimisation of the risk of unforseen side effects consequent upon its introduction strongly influence design decisions. This means that the design of changes to the primary distribution network is a task characterised by deliberation and conservative attitudes. This influences the professional norms and working practices of the designers who do this work. Because the consequences of realising changes to the installed network which are a consequence of a flawed design are so great both in economic terms and in operational ones, professional practices can be seen to have been adopted to minimise the possibility of design errors. The form of presentation of a design proposal, the planning report, can be understood in this light. Planning proposals are described in detail later but here it is worth mentioning that the planning proposal, as well as containing a description of the design solution proposed, also contains a summary of the major possible contending alternatives and a rehearsal of other technical and financial considerations which support the case for the design proposed. Evaluation of alternative design solutions, comparison with alternatives and justification of design proposals are a major influence on the design practices chosen for study.

1.3.2 Specific examples studied

The designs which formed the specific subject material of the case study represented about eight years' design work for the designer interviewed. The elapsed time between the start of a design and its completion varies between one to three years. Several design projects are therefore underway simultaneously and the designer is often occupied with other work related to his wider job responsibilities, for example, he is required to make his expertise available to other specialists within the organization fairly regularly. Eight distinct design projects were the subject of the majority of the investigation although, at times, the designer made reference in passing, for illustration, to other specific design experiences. The capital cost of realising these eight designs varied from one to two million pounds for the smallest up to twenty million for the largest.

1.4 Researcher's Background

The researcher, a chartered electrical engineer, has formal training in the electricity supply industry and several years past experience of working as a professional engineer in an Area Electricity Board (although not in the REC where the study was performed). Opinions in the literature differ as to whether or not it is an advantage for a knowledge engineer (the researcher's role) to be familiar with, or even a practitioner in, the domain being studied. The main disadvantage cited for this can be summarised by the adage " a little knowledge is a dangerous thing", which may result in the knowledge engineer being inclined to make assumptions or to take short cuts unconsciously, paying too little attention to the "real" expert

under study. At the other end of the spectrum of opinion, some researchers have suggested that the only way to effectively overcome the communication difficulties and the bottlenecks in knowledge acquisition is to make knowledge engineers out of experts, cutting out the role of "professional" knowledge eliciter altogether.

In the case of the work presented here, it was found that the researcher's credibility was greatly increased in the opinion of the expert designer as a result of their shared background of training and general electricity supply industry experience (to the extent that it is believed by the author that the case study as it is presented would not have been possible otherwise). It was also possible to progress more rapidly to what was of interest i.e. the designer's particular expertise, because of the reduced need to explain general organizational matters and basic technical terms. The researcher remained alert to the possible pitfalls of over-estimating a priori shared knowledge. The aids used for knowledge elicitation described below in section 3 helped to avoid some potential problems by the means they provided for some verification of the knowledge elicited. There were a few occasions when the expert over-rated the extent of the knowledge engineer's familiarity with technical matters and operational issues. On balance, however, on the basis of this case study the statement by Hogely and Korncoff (Hogely, 1986) that,

> "Each engineering discipline is replete with jargon and concepts that are entirely foreign to someone without an appropriate engineering background. A knowledge engineer who is not also well acquainted with the engineering discipline will be unable to perform the necessary critical analysis of the knowledge." (p.1158)

can be agreed with and further that their preferred solution to this dilemma, namely to use a knowledge engineer who has sufficient knowledge of the domain to communicate with the expert to be able to view the expertise critically, proved to be apt in this case.

2 Knowledge Elicitation - Raw Sources

The primary source of information was a series of interviews held with the design engineer. Copies of the designs, in the form of the planning proposals submitted for approval to the appropriate authorities within the REC, were also used for reference.

2.1 Interviews

Ten meetings were held with the designer. Each of these lasted approximately one and a half hours usually one hour of which was conducted formally and tape recorded. The first meeting (interview 0) was entirely informal and was used to explain the purpose of the research, to discuss how to proceed and to agree mutually satisfactory arrangements for the subsequent interviews. The final meeting (interview 9) mainly

consisted of a site visit to see the construction work underway to construct one of the major designs which had been discussed during some of the interviews. This formed a satisfactorily rounded conclusion to the series of meetings. Interviews 1 to 7 were conducted within a three month period; interviews 8 and 9 were carried out six months later to follow up and conclude the earlier work.

Each recorded interview was transcribed verbatim or by a mixture of detailed notes and transcribed passages where that was more appropriate. Subsequently, each transcript was partitioned into numbered "sections" for easy reference and analysis. A section is a coherent comment or fact on a particular topic so that a section may be as little as a phrase in a sentence or as much as an exchange occupying a page of transcription. In this thesis direct reference to interview material is made by giving the interview number as Int1 - Int9 followed by the section number e.g. Int2.14 meaning section 14 of the second interview. The techniques used during the interviews and between interviews to analyze the data are described in detail in section 3 of this chapter. However to give an overall impression of what went on a brief summary of what the interviews addressed is presented here.

OVERVIEW OF INTERVIEWS

1 The designer gives a general description of the procedures from initial "problem" arising, through exploration of solution possibilities, to final authorization of a planning proposal. He uses a specific example to "talk through" the process. (Transcribed verbatim.)

2 Firstly there is a discussion of questions arising from analysis of interview 1. The designer talks through a particular design situation choosing a simple one, covering initial problem presentation through to presentation of proposal for authorization, paying attention to the alternatives considered and presented in the planning proposal. (Transcribed verbatim.)

3 The interview begins with discussion of questions arising from analysis of interview 2. This is followed by discussion of the influence on design of security of supply standards and the policy for primary system development in general; how to establish the network context relevant to a design; and the generation and presentation (in reports) of alternative designs considered. The designer uses a more complex design example (than used in interview 2) for illustration. (Transcribed verbatim.)

4 The repertory grid exercise is begun (see section 3.2 below) using six design examples. (Notes and some fragments transcribed verbatim, cards from grid exercise.)

5 The repertory grid exercise is completed using a draft grid drawn up after interview 4. (Notes and verbatim transcribed fragments.)

6 A check back using constructs from the repertory grid exercise to confirm descriptions of the designs used for the exercise is made. There follows discussion of a design situation for which there are many (> 6) plausible alternative forms of solution including discussion of how these are grouped and selected for presentation in the planning report. (Transcribed verbatim.)

7 The designer talks through issues arising in a very large and complex design proposal. (Transcribed verbatim.)

8 (After long interval since interviews 1-7.) The interviewer gathers basic factual information about the organization and the designer. Some points arising from earlier interviews are checked. A repertory grid check is conducted - checking understanding of the constructs by presentation to the designer of a description of a design *not* included in the grid exercise. (See section 3.2 below for further detail.) Discussion of design commitments resulting from analysis of material from earlier interviews follows. (Notes with some passages transcribed verbatim.)

9 There is a fairly short informal session prior to a site visit. (Notes on relevant parts of the conversation with some passages transcribed verbatim.) This is followed by the site visit (photographed) to see the realization of the design discussed mainly in interview 7.

2.2 Documents

Complete copies of the planning proposals for the eight distinct projects which formed the basis of most of the discussions were provided for reference. These were read after discussions about the designs to which they referred and they were always treated as supplementary, supporting material. The pitfalls of using factual information drawn directly from reference texts or other written documents without mediation to show how "facts" or "data" contained within them are used by an expert at work have been acknowledged for a number of years. Knowledge engineers have been counselled against direct use of "text book knowledge" even since the days of first generation expert system construction (Hayes-Roth , 1983). (McDonnell (1986), for example, is one of numerous studies which rehearses some of the problems.)

Planning proposals' contents vary but there are structural similarities among them. A typical proposal might contain the following : an executive summary; an introduction to the salient factors giving rise to the proposals contained in the report; a description of the existing electrical network affected by the proposals; (optionally) extra sections dealing with contextual aspects which are particularly pertinent and therefore worthy of separate elaboration e.g. system loading or security considerations; a characterization of "the need for action"; descriptions of at least two, but typically three alternative proposals dealt with in some detail; a review of the engineering considerations; a review of financial considerations; and finally a section of recommendations. Each proposal also includes some maps showing the geographical location of substations and the extent of substations' supply areas; substation layout diagrams; schematic diagrams of parts of the electrical network; and tables of relevant figures e.g. comparing expenditure, showing network load forecasts, or providing fault statistics.

A copy of the published standards relating to security of electricity supply (Electricity Association, 1978) to which the REC conforms has also been referred to directly. Reference to this was mainly for the purposes of checking what information is publicly available so that sample material published here from the case study could be chosen to avoid any concerns about confidentiality. However, it should be noted that the interpretation of this standard for security of supply which is given in chapter 7 is primarily based on the designer's references to it during the interviews.

3 Knowledge Elicitation - Aids Used

In this section systemic grammar networks and the repertory grid technique used for elicitation and analysis of elicited material are described. Both were used, in different ways, to direct the discussion during

the interviews, to represent outcomes and to test the interviewer's understanding. In the accounts which are given, in each case the aids are described, the way they were used for this case study are explained and an evaluation of their appropriateness is given. Section 3.3 gives similar but more brief treatment to the grounded theory approach to qualitative data analysis. This approach was used in a very limited way for data analysis but the theory building aspects of it played no part in this case study.

3.1 Systemic Grammar Networks

Systemic grammar networks derive from the school of systemic linguistics originating in the work of the structuralist Saussure (Saussure, 1916). However their roots lie more specifically with Firth (Firth, 1957) who studied the contextual features relevant to the practical functions of language in use. Halliday (Halliday, 1978) developed Firth's ideas into a linguistic theory of the social interpretation of language and meaning. In Halliday's terms, in order to make sense of what the speaker actually says, "we have to interpret it against the background of what he 'can say' ... it is the actual seen against the background of the potential" (op.cit., p.40). A network represents the potential, the network of options and the inter-relations between them. A note outlining the origins of systemic grammar networks in more detail appears as an end note to this chapter.

3.1.1 What systemic grammar networks are

Systemic grammar networks (SGNs) are a way of representing functional grammars. Nodes in the network represent terms in the grammar and a variety of links represent the relationships between the terms. Halliday's systemic grammar is a theory of linguistic analysis in which languages are described as systems that allow speakers to compose and to interpret utterances in the context of a situation. A central idea in systemic grammar is that meaning in language is associated with differences - a word or phrase - a term, derives meaning from the context given by contrasting it with the choices, the alternative possibilities from which it is selected. Bliss (Bliss, 1983) and co-workers have advocated the use of systemic grammar networks as a means of analyzing and representing qualitative data and coping with the complexity of classifying it. They recommend the use of SGNs in situations where individual prose accounts are not appropriate but where, on the other hand, categorization is not a simple, or straightforward matter. In the analysis of qualitative data using SGNs linguistic analysis is not attempted rather the analyst *constructs a language* to suit his purposes. The SGNs show the terms of the language (the categories chosen) and how they relate to one another, so what is transferred from linguistics is the idea that meaning is given by making contrasts in a given context.

Networks are a simple extension of the idea of putting things into categories. A network shows the categories chosen and how they relate to one another.

"To categorize is to draw distinctions and to name them, recognising that distinctions may need to be drawn along several independent dimensions, and that any distinction may need to be further divided into subsidiary divisions. Networks offer a uniform notation to express such schemes at any required level of complexity, and a terminology intended to clarify and assist communication of the issues involved." (Bliss, op.cit., p.10)

In a SGN, category names (the terms) appear as nodes in the network at all levels. The links between terms specify the possible choices among them. The symbols for links shown in figure 6.1 have the following meanings. The two simplest elements of the network notation support representation of mutually exclusive sub-categorization of a larger category (a) and co-selection of a number of distinct and independent aspects of a category all of which have to be represented (b). A simple categorical description is formed by passing once through a network composed with this notation. However, the notation is enriched by its ability to represent recursion. Recursion expresses the fact that exclusive categories may sometimes be applied in combination (c) and that in co-selection there are several aspects of something which may need to be described (d). Finally, the network notation can also accommodate categorical distinctions which only apply when restricted entry conditions are satisfied (e).

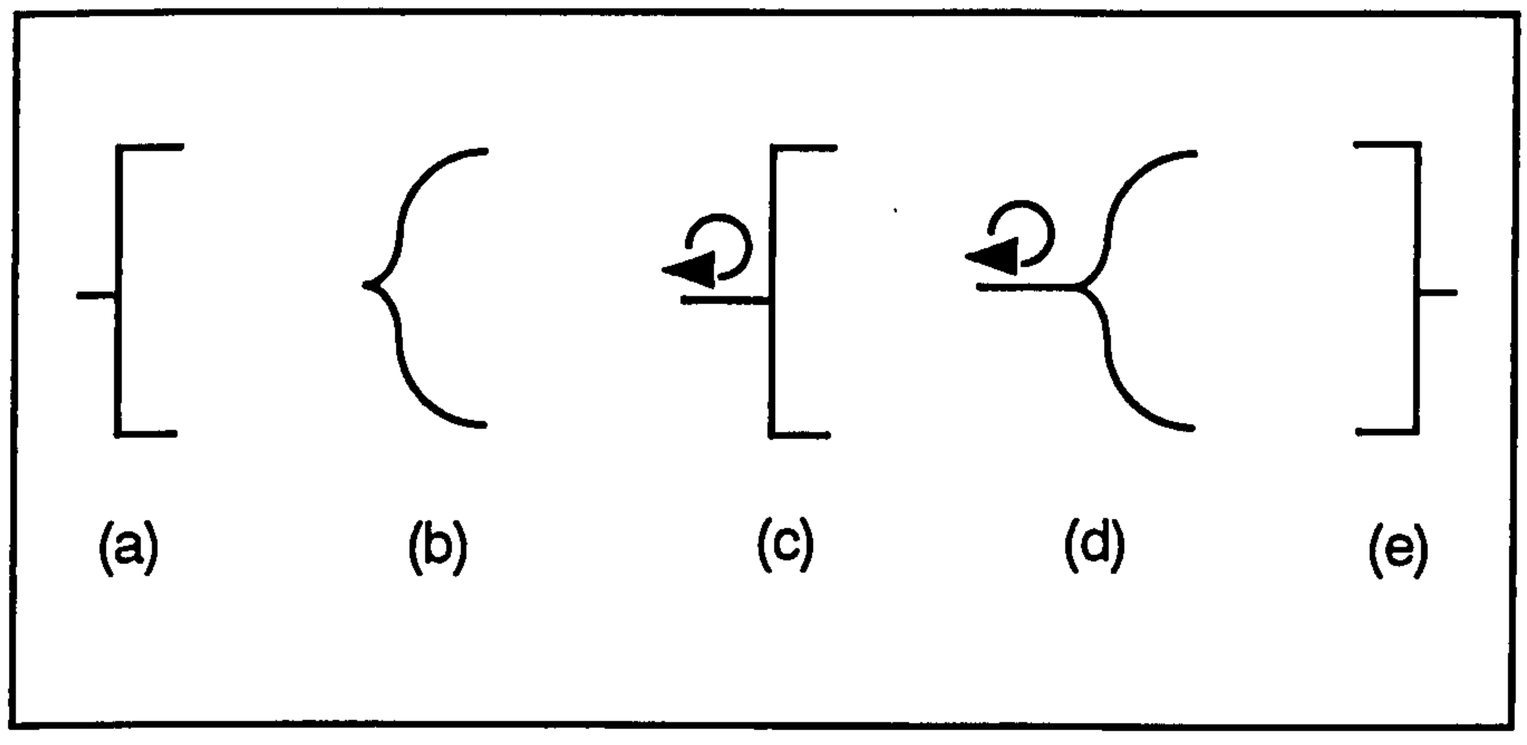


Figure 6.1 SGN symbols

A network represents a view of the qualitative data it describes. The notation does not impose a choice of terms or a refinement of them onto the analyst. These matters of decision and judgement are left to him or her. The network merely displays the outcome. In the same way, it is left to the judgement of the analyst as to what is relevant, for example the point beyond which no distinctions are to be made.

One of the major strengths of a network is the way it can be tested. Each path through a network should produce a categorization which is plausible and which makes sense. The network, if correct, should constitute a finite

set of allowable descriptions, no possible description should be nonsense. A network is a compact way of expressing a large number of finely differentiated descriptions organized so as to show the similarities and differences, that is the relationships between the terms.

"To construct a network and to use it to encode data is like constructing an artificial language, which offers meanings and distinctions of the kind one wants, and then using it to give an account of data in those terms." (Bliss, op.cit., p.27)

SGNs have been used in educational research to represent, for example, problem solving by chemistry students, a child's knowledge of mathematics, and the categorisation of students' comments on their peers (all of these are described in Bliss, op.cit.). Johnson (Johnson, 1985) has suggested the use of SGNs as a mediating representation in an expert systems context - as a way of analyzing and representing knowledge in an implementation independent form thus "mediating" between verbal data and computer based knowledge representation schemes. Some examples of applications which have used SGNs in this way are: the characterization of the design decisions of a VLSI chip designer (Johnson, 1987), the representation of the task of estimating the cost of manufacturing engineering components requiring multiple machining operations (Al-Shawi, 1990) and as an aid in eliciting knowledge used by designers of page layouts (Tunnicliffe, 1990).

3.1.2 How SGNs were used

SGNs were constructed to analyze and represent the data from the interview sessions. The conceptualisation and abstraction which resulted from the process of constructing SGNs between interviews was checked with the designer at following interview sessions by prompting questions which formed the basis of further discussion with the designer and by providing material for generating descriptions which were used for "teachback" within the interview sessions. Teachback, grounded in the ideas from Pask's conversation theory (Pask, 1974) actively involves the interviewer in explaining back to the expert using the expert's own terms what has been understood, and thus verifying, through the judgement of the expert, the reconstruction that has taken place. This technique is described more fully in Johnson (1987) where it is set in relationship to conversation theory in the context of interviewing specifically for knowledge elicitation. The link between conversation theory and the approach to knowledge elicitation reported here is a direct and important one which is dealt with in section 4 of this chapter.

The process of constructing and refining SGNs throughout the interview series is illustrated on a small scale by the fragments of SGNs shown in figures 6.2 and 6.3 and the accompanying relevant annotated extract from one of the interviews which is shown in figure 6.4. The first SGN fragment (figure 6.2) shows a portion of the network which prompted the discussion shown in figures 6.4a and 6.4b whilst figure 6.3 shows the revised fragment

of SGN resulting from the discussion.

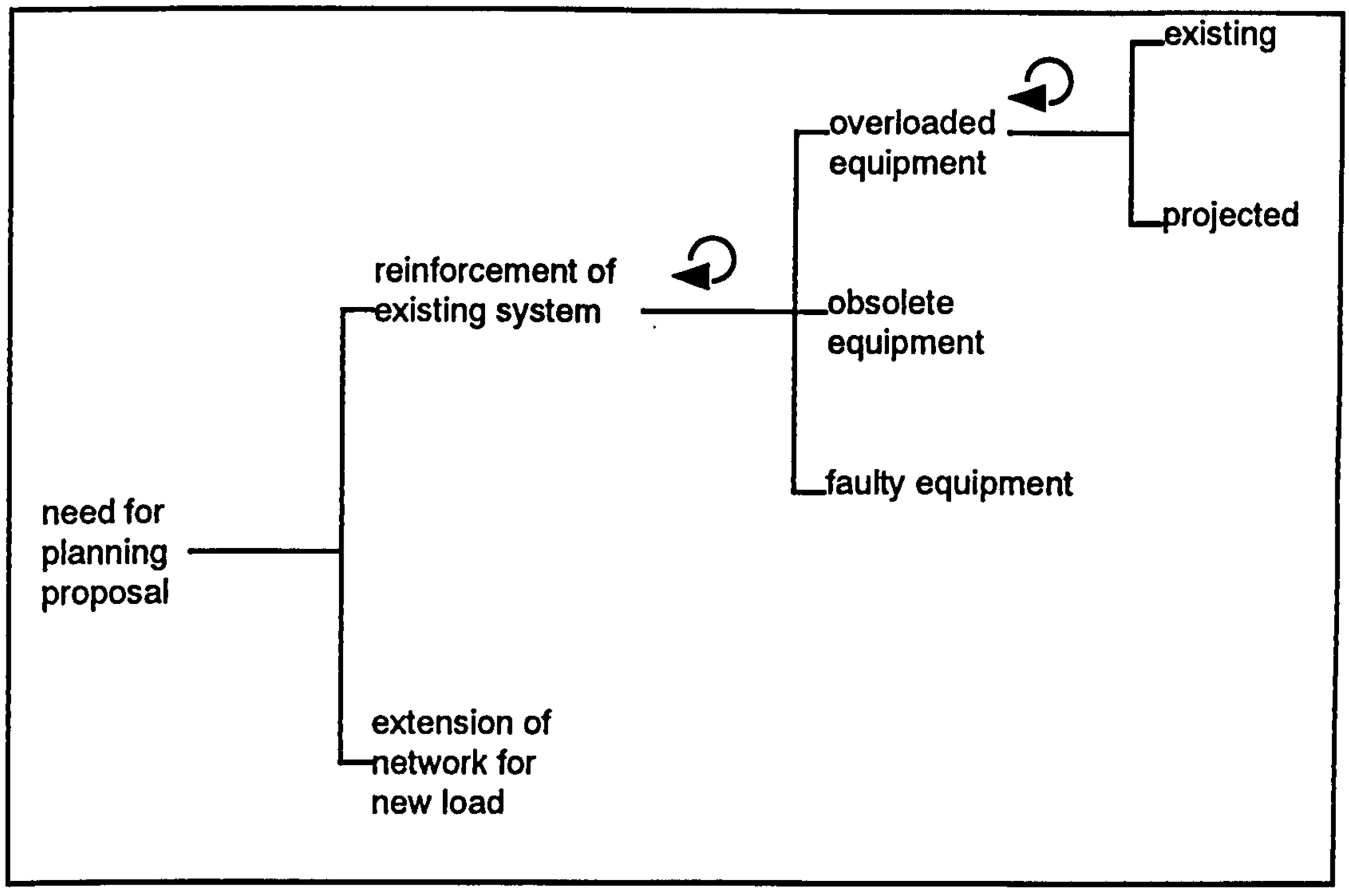


Figure 6.2 Fragment of a SGN produced during early data analysis, prior to the discussion show in figures 6.4a and 6.4b.

Two formal interviews have taken place. The interviewer has begun to produce a number of systemic grammar network fragments to represent different aspects of the design situation. The interviewer is seeking to validate the network fragment, shown in figure 6.2, with the designer but it does not seem to fit a specific design proposal used by the designer to illustrate his points.

Interviewer: [this design proposal] is described as reinforcement ... but it's about a new substation [represented in the systemic grammar network as extension of the network for new load] being put in ... is there any distinction between reinforcement and extension? When would you describe something as system extension? ... and when is it a system reinforcement?

The designer answers, clarifying the term "reinforcement". The interviewer comes to recast the notions the designer has been using when identifying the initial causes for design proposals.

Designer: ... we divide the project into either reinforcement or replacement, we've only got the two categories ... we would automatically translate system extension into system reinforcement ... we're looking at one and the same thing.

Interviewer: right, we've just got reinforcement and replacement.

Designer: ... we have a category system for deciding which classification capital expenditure falls into ... and basically the two broad groups are replacement, oh, sorry, there's three broad groups, there's really replacement, and reinforcement which can subdivide into load-related reinforcement where you've got new load coming up and you do a reinforcement, or if you like, your system extensions, and reinforcement which is purely where you haven't an identifiable new development say and it's just a general mass of load growth.

Figure 6.4a Annotated extract from transcript of third interview.

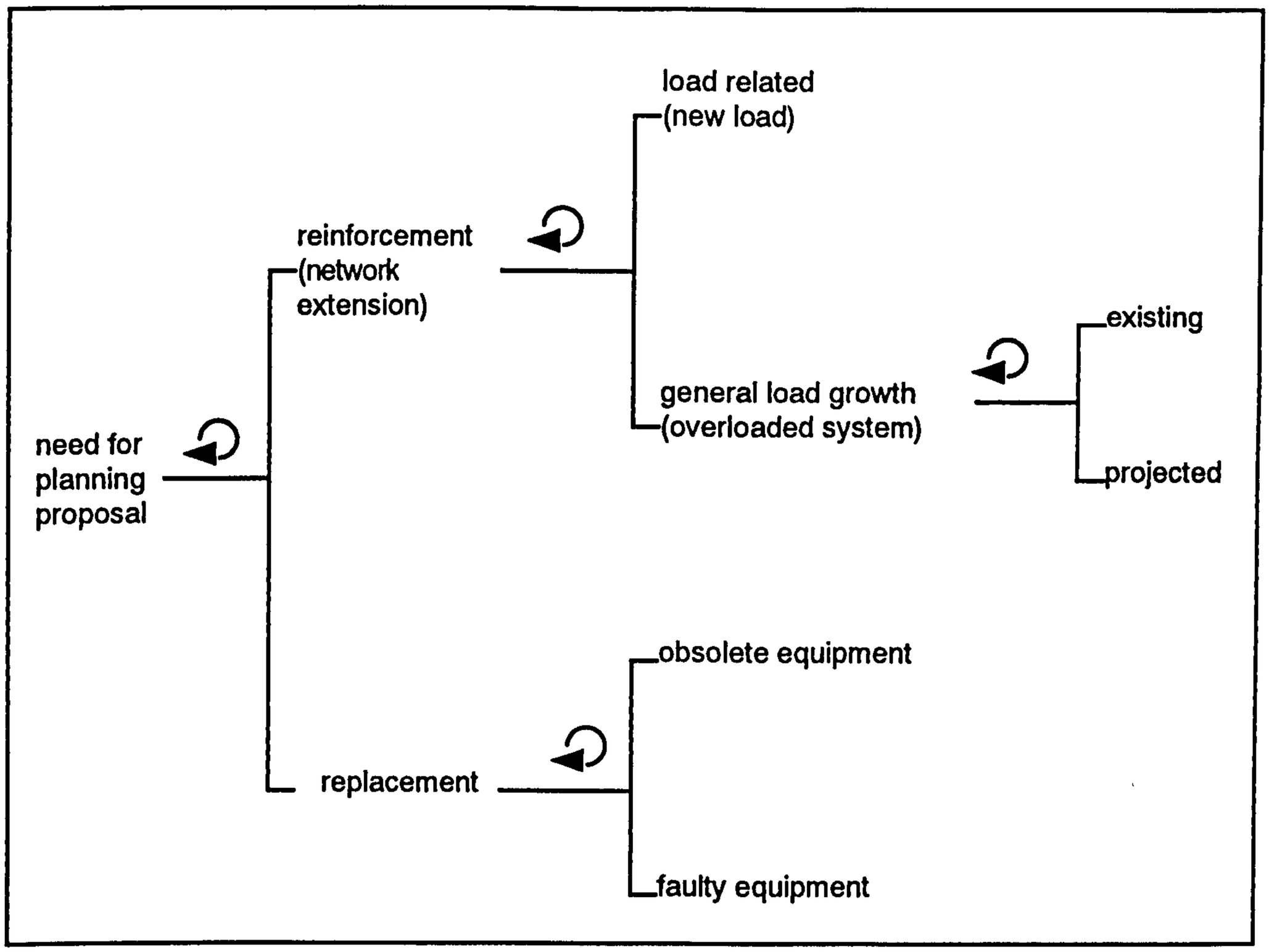


Figure 6.3 Revised fragment of a SGN after discussion shown in figures 6.4a and 6.4b.

One advantage of SGNs is their content-free nature. This was turned to advantage by using them to represent different concepts. For example SGNs were drawn to represent a number of aspects of the designer's task, the design process and the overall context within which a particular design is evolved. These SGNs, among others, are shown in chapter 7. The complete set of SGNs represent: the overall design context, the initial design requirements, the (electrical) network context relevant to a design requirement, some of the main sources of constraints operative in a design situation, and the range of design alternatives open to the designer. The other strengths and benefits of using SGNs are now considered.

The interviewer changes her view. She tests this new view by using the designer's terms to describe paths through the systemic grammar network which classify two design proposals: she checks the notions of reinforcement and replacement with the designer using a second design proposal and he expresses his agreement with her view.

Interviewer:	right, what about these two schemes we've got here [referring to two design proposals]. This [design proposal] is a reinforcement ... and this [design proposal] is a replacement ... so [we see some of] the reasons why you might be replacing, what about if you've got an overloaded system? ... that's reinforcement.
Designer:	that's REALLY reinforcement.
Interviewer:	... whereas obsolete and faulty [equipment] is ...
Designer:	... is replacement.
Interviewer:	even though they might actually provide [i.e. result in] an enhanced system

Finally, the designer reveals that few design proposals arise from a single identifiable requirement.

Designer:	oh, yes, and then everything tends to become - and this is the problem, if you like, with all the schemes that we've got - there's very, very rarely a single identifiable reason for doing it.

This data prompts a change to the left-most bracket in the systemic grammar network fragment. [The revised fragment of network is shown in figure 6.3 - compare figures 6.2 and 6.3.] The design proposals under discussion are unusually simple ones chosen by the designer to allow uncomplicated illustration of points early on in the interview series. More complex designs are included in later elicitation sessions.

Figure 6.4b Annotated extract from transcript of third interview.

3.1.3 Main strengths and benefits

The use of SGNs allowed analysis of the verbal data and representation of knowledge to be carried out independently from implementation and the issues associated with implementation, i.e. in a way unconstrained by a knowledge representation formalism. Other researchers have made similar observations:

> "The (systemic grammar) network appears to be most useful in the early stages of the knowledge elicitation cycle. During our initial interviews, we found any mention of rules merely served to antagonize the experts. We think this is because imposing low level knowledge representation schemes in the early stages of the knowledge elicitation cycle limits the conceptual richness of the elicitation dialogue. Our SGN approach gave us the freedom of conducting knowledge elicitation sessions with no strong epistemological theory, and served to focus the experts' attention without constraining them." (Al-Shawi, op.cit., p.144)

SGNs were a useful aid during the elicitation process enabling an interpretation of the verbal data to be formally represented. Since SGNs were constructed and refined between interview sessions they became a valuable guide to the elicitation process. This role is demonstrated, although with a rather simple example, in figures 6.2, 6.3, 6.4a and 6.4b shown above. The generative nature of SGNs was exploited by using

with the designer. This validation process was itself a valuable technique for further knowledge elicitation as illustrated in the figures mentioned.

The ability to verify the knowledge represented in the SGNs allows checking and refinement to take place prior to implementation thus allowing additional iteration steps early in the model building process (i.e. earlier than even a prototying approach permits). This is particularly valuable in the early stages of knowledge elicitation when conceptualisation of the area of investigation is taking place. It avoids the "be-littling" of the expertise which might otherwise occur if superficial, isolated "rules" are implemented directly early on while they are still (in the understanding of the knowledge engineer) divorced from the complex situational context within which they might conceivably occasionally be true.

In the case of the designer's expertise the complexity and variety of design issues which concern him can be shown clearly and compactly using SGNs without recourse to the artificial over-simplification (typically hierarchical decomposition) which is commonly resorted to otherwise. SGNs therefore have something to offer from both a structuring and a documenting perspective. Throughout Part 1 of this thesis the case has been made for viewing design as a holistic, solution-oriented activity in which design considerations are highly interconnected. Tunnicliffe, having come to a similar view of design, suggests that the ways design knowledge is elicited must take this nature of design into account and should therefore take place using holistic strategies.

> "The high-level strategic knowledge and knowledge structures used by designer experts exhibit complex interrelationships that are probably not decomposable. Thus, although explanations of the nature of sub-task or component level interactions may be provided by the expert designer, they are unlikely to be accurate models of the design activity. ... Consequently, reductionist philosophies of knowledge elicitation are less appropriate for design applications because design is intrinsically more dependent on a subtle, but nevertheless powerful, contribution of knowledge that is altogether more holistic in nature." (Tunnicliffe, 1992, p.29.13)

The terms which appear in the SGNs are derived from the data, an SGN should characterize all the terms which are relevant and only those which are relevant to what is being represented. There is an important methodological significance in this point concerning setting the boundaries for so-called "deep knowledge" which has already been discussed (chapter 4 section 1.5). It is incumbent upon the knowledge engineer to keep this firmly in view.

> "It is the task of the knowledge engineer to extract from the data those aspects of the knowledge which are relevant to the task in question. Invariably there is a great deal of information either in the data or in text books which is either not used by the experts or which lies outside the scope of their considerations. For instance, there is a wealth of

formal and empirical theory associated with the (metal cutting) process. Estimators (whose expertise is being elicited), however, do not consider such a level of detail in their tasks. Those aspects of the domain which are relevant to their tasks must be brought out in the network if it is to successfully characterize their knowledge. The boundaries of the knowledge in the final system will be set by its proposed functionality and its operational environment as well as the nature of the experts' knowledge. Since these considerations should guide the knowledge elicitation process in general, they should also be reflected in the construction of the network." (Al-Shawi, op.cit., p.143)

Some reports of using SGNs for knowledge elicitation, including that of Al-Shawi (op.cit.) claim success with using SGNs directly to communicate with their interviewees. In the case study reported here, diagrams of networks were not presented to the designer directly but were used by expressing their content verbally, e.g. for generating examples. This is achieved easily and in a natural manner directly from SGNs since they do, after all, represent functional grammars.

Other investigators have suggested that SGNs might be useful for communicating and discussing the outcome of knowledge elicitation among members of a knowledge based system building team where the project is too large for a single knowledge engineer to operate alone (Johnson, 1990). Tunnicliffe and Scrivner (Tunnicliffe, 1992) report success with the use of SGNs and the teachback technique to structure, record and transfer to an implementation team the results of design knowledge elicitation. The case study reported here offered no scope for testing these ideas, however it was found that the SGNs constructed did prove to be a useful permanent record of the structure (grammar) of the knowledge from which selection of knowledge to be represented in an implementation could be made at a later date.

3.2 Repertory Grid

According to the theory of personal construct psychology (Kelly, 1956) constructs are ways of construing the world, enabling people to respond in ways which are "explicitly formulated or implicitly acted out, verbally expressed or utterly inarticulate, consistent with other courses of behaviour or inconsistent with them" (Kelly, op.cit., p.9). Kelly's repertory grid technique was originally developed for use in clinical psychology as a technique for exploring individuals' personal constructs about interpersonal relationships in the context of psychotherapy. However the technique has been applied extensively to the elicitation of personal constructs for other purposes. Thomas and Harri-Augstein (Thomas, 1985; Harri-Augstein, 1991) have used it as a central component of "grid conversations" which are the basis of their conversational science of self-organized learning. It is for its value on this theoretical basis, i.e. as a tool for conversation (elaborated upon in section 4 of this chapter) rather than from Kelly's original claims (that psychological events are real phenomena) that the repertory grid technique has been used here.

3.2.1 What The Repertory Grid Technique Is

The repertory grid technique is a content-free procedure for exploring and for forming what are known as personal constructs. The notion underlying this technique is that humans can represent their environment - the situations with which they are faced - by placing alternative constructions upon them. In the case study reported here the repertory grid technique was used as a tool for *encouraging reflection*, in the spirit in which its use is advocated by La France (La France, 1990) i.e. as an aid in exploring how expertise is organized in terms of abstract as well as surface features (La France, op.cit., p.60).

Applications of the repertory grid technique for business systems analysis (Stewart, 1981) and for studying management decision making (Shaw, 1980) have been reported over a number of years and a few reports of its use for requirements elicitation have begun to appear (Gutierrez, 1987). Shaw and Gaines (Shaw, 1987) have described how the elicitation of personal constructs using the repertory grid for knowledge elicitation purposes can be supported by an interactive computer program. A repertory grid exercise can be used "to develop the expert's vocabulary ... by encouraging him to make clear the distinctions he uses in applying his expertise" (Shaw, 1987, p.110). In this context the repertory grid represents the personal constructs used to make *relevant* distinctions between elements, in this case study, to distinguish between designs.

A construct is a way in which some things are construed as being alike and yet different from others. It is therefore inherent in the nature of a construct that it is bipolar. The personal aspect is an important part of the repertory grid technique since personal constructs, i.e. those relevant to the designer, are the ones which indicate the way he classifies his experiences. These constructs give the dimensions of personal meaning, the poles of the constructs are the limits of the dimension.

The process of eliciting personal constructs from a "subject" using the repertory grid technique proceeds by choosing *significant* items of experience. These determine the scope of the ensuing conversation. The purpose of the grid exercise needs to be clear at the start so that the type of elements that will best allow the purpose to be achieved can be identified. Design proposals that the designer had been personally responsible for producing were used as the items of experience in the work reported here since it is important with this technique that each item chosen is one with which the subject is familiar and which is meaningful to him. The chosen items of experience constitute the set of elements to be compared and contrasted with one another in the repertory grid exercise with a view to eliciting a set of useful constructs. Any distinction that is important to the subject is a valid construct. Elements are grouped into threes (triads) since in its minimum context a construct is a way in which at least two elements are similar and contrast with a third (Kelly, op.cit., p.61). The subject is asked to compare the elements in the triads, to consider their similarities

and differences and to describe them.

Constructs are elicited by considering different combinations of three elements until no more new constructs seem to be emerging and a good cross-section of the possible combinations has been considered. A "raw" grid is drawn up in which elements of personal experience head the columns and in which rows denote the constructs. Descriptions of the poles of each construct which are deemed satisfactory by the subject label each end of a row. The grid is subsequently "focused" by encouraging the subject to assign each element to each construct by indicating to which poles each can be assigned or to which it has more tendency to be associated. Elements which are assigned to similar poles can then be clustered together to display a pattern of personal meaning. Elicitation continues throughout grid focusing by including the subject as an active participant in the activity.

> "Thus by exploring the clustering of elements and how they have been assigned to the pole descriptions (of clusters of constructs), and by studying the clustering of constructs which have separated out clusters of those elements which are assigned in much the same way, it is possible to reflect the unappreciated patterning in a client's feelings and thoughts about the topic back to them for more serious consideration." (Thomas, op.cit., p.68)

Conversation based on focusing a grid allows the constructs to be refined. If a good representative selection of elements (items of experience) have been used for the grid each construct should finally represent an important dimension of the subject's construing (Thomas, op.cit., p.78). In short, the repertory grid expresses something about the way a person looks at things. In this capacity it can be used as a powerful conversational technique without commitment to theories about construct systems or any possible structures underlying them.

3.2.2 How The Repertory Grid Technique Was Used

A repertory grid exercise was carried out in two sessions - during the fourth and fifth interviews. In the first session elements grouped into triads were compared to elicit the construct poles. These were formed into a raw grid. During the second session the raw grid was filled in and focused in the way described above (section 3.2.1) resulting in refinement of the constructs and the eventual production of something approaching a focused grid. In other words, a conversation was held between the designer and the knowledge engineer in which descriptions of the poles of the constructs were agreed. (Appendices 1 and 2 are detailed records of what went on during these two sessions.)

Six design proposals were chosen as the elements for the exercise. Seven design proposals had been made available by the designer. However, only six of these were chosen, partly to make triad combination formation

easier, but primarily because one of the seven proposals was far larger and more complex than the others (as an indication of this it was of an order of magnitude higher in capital cost than any of the others), thus being an exceptional case, it was excluded from this exercise. It might be thought that, ideally, more than six elements should be used for a grid exercise. However an over-riding consideration is that the items should be items of *personal experience*. The elements must be personally significant to the one from whom constructs are being elicited. The planning proposals which were used were all the results of design work which the designer had personally carried out or had directed closely and each represented problem situations and design solutions with which he was very familiar. The "personal construing" aspect of the exercise was emphasised to the subject before commencing the exercise so that he should be discouraged from filtering out anything that occurred to him on a mistaken assumption about what sort of constructs were appropriate (see appendix 1 - Int4.3).

In preparing for the exercise, the knowledge engineer grouped the six proposals into different combinations of three. The intention was to examine different combinations of three from the six until nothing more which was useful (i.e. different) seemed to be emerging. This was explained to the designer. Notes were made on cards as the exercise progressed to keep a record of what emerged. The designer was asked to comment on similarities and differences between the proposals, three at a time as described in 3.2.1 above. To assist in this process copies of the design proposals were physically placed in front of the designer three at a time for consideration. Whilst the designer rarely had to refer to the content of a proposal their physical presence aided him in bringing them to mind for comparison with each other. Constructs were elicited by this process, they ranged over the initial design requirements giving rise to the proposals, the nature of the design solutions proposed, and the alternatives that were considered in arriving at a solution. Emphasis was placed on eliciting constructs which described the kind of solutions proposed i.e. their qualities (see appendix 1 -Int4.4).

A raw grid was drawn up as a result of the first session (interview 4). The second session (interview 5) was used to explore the constructs elicited in the first session and to refine them. Discussion was focused on filling in the grid by identifying the poles of the constructs which best applied to each element and through this activity refining the constructs where necessary i.e. modifying the pole descriptions to better describe the designer's notions. Some constructs did not apply to some elements, these were noted. The grid resulting from this second session is shown in appendix 3.

Data elicited during the repertory grid exercise was used to continue development and refinement of the SGNs. There was some evidence that the repertory grid exercise brought out useful distinctions which play a part in the designer's work which would be difficult to elicit by other more superficial discussion. This point is discussed further below (in section 3.2.3) but here an illustration of how data from the repertory grid exercise was used to refine SGNs is given through a small example. The

comparison of one combination of three design proposals elicited the distinction that two were alike through involving equipment which had become obsolete due to its age whilst the third differed from the first two due to equipment becoming "unexpectedly" obsolete due to a manufacturing problem. It was quite clear from the subject's reaction to his own description of this distinction that it surprised him and was not something he would have brought out in a straight forward description of any of the proposals and he said as much at the time. The notes of interview 4 record this :

> "The subject clearly had an immediate inclination to make this split but took some time (silently) to think about why it had been made. The subject was clearly surprised at what he found out by this reflection." (appendix 1 - Int4.25)

The analysis of the data from interview 4 resulted in addition of terms of increased delicacy to one of the SGNs transforming the previously terminal term "obsolescence" as shown in figure 6.5 and giving rise to questions (see appendix 2 - Int5.17) about the use of the term "obsolescence" as applied to different objects (namely cables, switchgear, transformers). The eventual outcome of this discussion was a considerable refinement of part of an SGN concerned with initial design requirements.

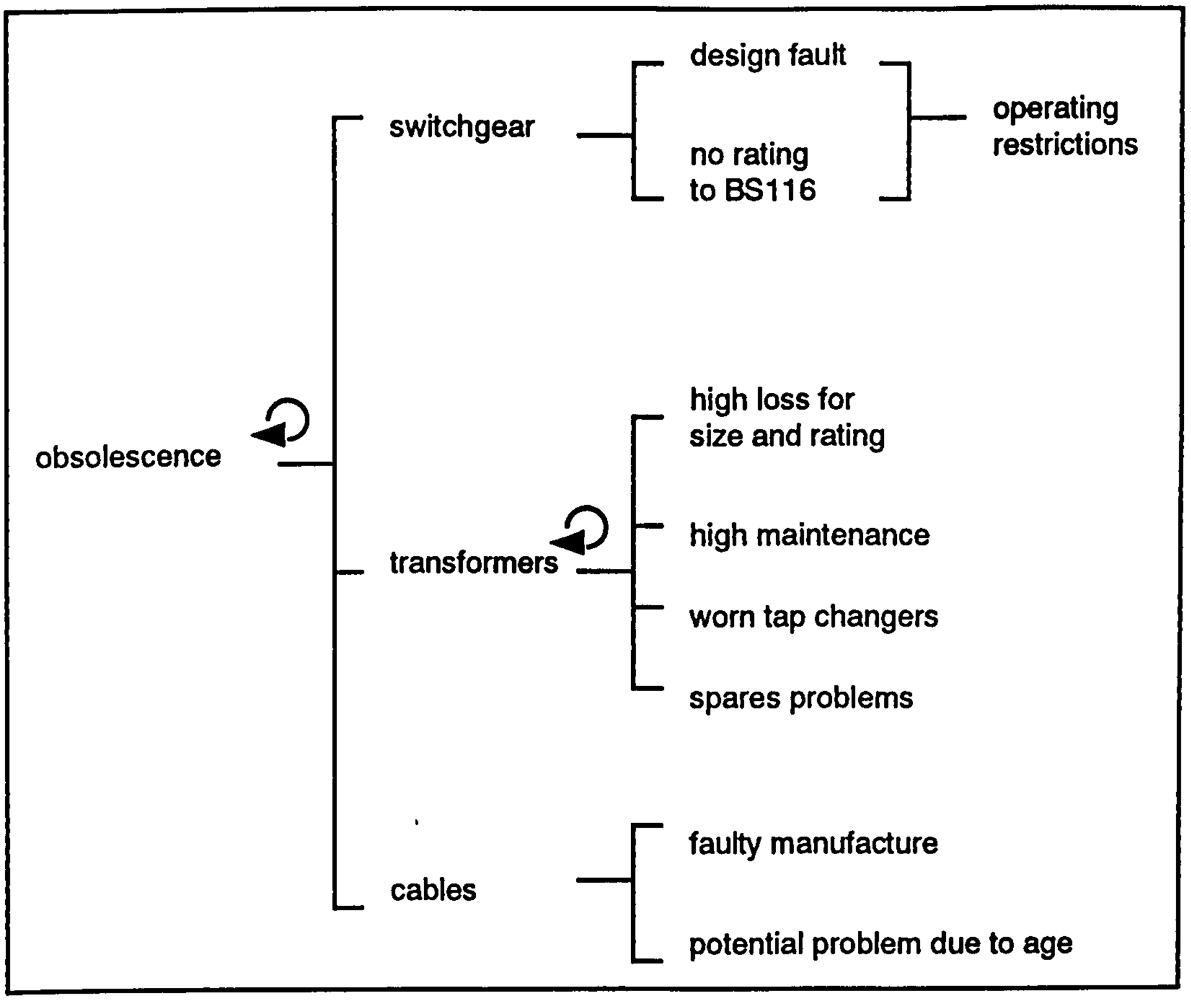


Figure 6.5 Term "obsolescence" refined following discussion arising from the repertory grid exercise.

Later in the interview series, during interview 8, a check of the data elicited using the repertory grid technique was made. A design proposal which had *not* formed one of the elements for the exercise was examined by the knowledge engineer. A set of statements describing this proposal using the constructs elicited was produced. The designer was invited to comment on each statement in terms of its appropriateness for describing the design proposal. Nine separate statements were produced with supporting descriptions. These are shown in figure 6.6. The outcome of this grid check is summarised in figure 6.7. This result was very encouraging. Scope for generating descriptions of design proposals for designs other than those included in the repertory grid exercise was severely limited for the reasons already given, essentially due to the size and nature of the designs being studied and of the working environment. However acceptance of the description generated by the knowledge engineer gave some basis for believing that the negotiation of constructs through the grid conversation had been successful - the knowledge engineer was able to describe a design problem and the solutions proposed in the expert's terms.

Transformer problems at Branse and proposed solution

(1) There is no appreciable load growth. There is no loading problem in this area.
The load has actually reduced over the last few years, the area is not one in which appreciable load growth is to be expected in the foreseeable future.

(2) The problem at this site is purely to do with obsolete components.
Main Problem: The transformers are sixty years old, the tap change mechanisms frequently break down and replacement parts have to be made specially because of the age of the equipment. The transformers leak badly, seepage of oil has become an embarrassment. Additional Weakness: Two of the four transformers are supplied from gas-filled cables of an age where they are expected to start to leak.

(3) Replacement of the transformers will completely solve the main problem, alternative solutions proposed will also completely solve the main problem and in some cases partially or fully address the additional network weakness.
Replacement of the obsolete transformers with second hand ones from elsewhere will solve the main problem. The replacement transformers will fit in with the existing network in that area in terms of age. The alternative solutions proposed which include the installation of new transformers (CER type) at a standard voltage with appropriate new switchgear, and other schemes which involve making use of auto transformers to secure the load in the event of fault would completely solve the main problem and would leave the network in that area better served by introducing newer equipment and/or by adding flexibility (through rendering the area less electrically isolated).

(4) Replacement of the transformers is a short term solution.
It solves the immediate problem and will last until there are further developments in the area or further appreciable degradation of the cables or other equipment necessitating major alterations to the system in that area. The additional weakness due to aging gas-filled cables is not alleviated.

(5) The other designs outlined are longer term solutions.
They offer more scope for development, more flexibility in the network for future development. Some of them eliminate the gas-filled cables, others reduce the problem, in others they remain in service.

(6) The problem at this site has been known about for some time.
It has not suddenly arisen, but has developed (increased) over a period of time.

(7) The work proposed will or has appeared in the capital programme.

(8) Replacement of the transformers is the cheapest solution. It does not lend itself to further development or offer extra longer term benefits.
There is no potential for expansion. Financial considerations (constraints) prevent justification of extra expenditure under the regulations currently in force relating to privatisation of the company. ("Claw back" and discount rate.)

(9) Description relating to security of supply - not printed for confidentiality reasons.

Figure 6.6 Description of a design situation to check grid constructs

Outcome of the Grid Check

Nine statements with supporting text were presented (see figure 6.6).

One of the statements (statement 9) was agreed but deemed to be too confidential to be published. It relates to security of supply issues.

The evidence or supporting text for statement 4 could have included more factors - the statement itself was considered correct as was the supporting text as far as it went.

One statement (statement 5) referred to "other designs outlined" i.e. in the unpublished planning proposal. To be strictly correct this statement should have referred to the alternatives discussed during the interview (which were more extensive) since several of these alternatives did not actually appear in the final report.

All the other statements were accepted with their supporting text without any qualification.

Figure 6.7 Summary of outcome of the check of grid constructs.

3.2.3 Main Strengths And Benefits

Useful distinctions emerge during a repertory grid exercise which are not obvious i.e. ones which are not superficial features that the expert could volunteer without some elicitation aid. Towards the end of the first repertory grid exercise session the expert remarking about a distinction he had just made, said, "That's an interesting point, this is an intriguing game ... it brings things out that I must confess I wouldn't have thought of" (appendix 1 - Int4.27). The exercise acts as a means for awareness-raising in the expert. This is not surprising as in some applications of the repertory grid, such as for self-organised learning (Thomas, op.cit.), this is a primary objective of its use. Because interesting distinctions emerge, an exercise using the technique sustains the interest of the expert and leads to high motivation to participate. It can be difficult to get the exercise started. An air of confidence on the part of the knowledge engineer is needed at the start as the formal consideration of each triad can seem daunting initially (for both parties). The experience in the case study was that the expert expressed reservations prior to the exercise but had sufficient confidence in the knowledge engineer to agree to suspend his scepticism and to give the exercise a chance. At the start of the exercise he expressed anxiety that he would run out of material (see appendix 1 - Int4.5) and confessed to be holding back a little i.e. "saving" distinctions for later use. However, he very quickly warmed to the exercise particulary once he began to see interesting notions emerging and quite soon was evidently actively enjoying it, he was soon "taking over" the process of moving from triad to triad using

the combinations prepared by the knowledge engineer (see appendix 1 Int4.5; Int4.8). After the third set of triads he was commenting,

> "I see what you're doing now, I see how you're exploring this ... I literally am playing all sorts of games with this ... you can play these in all sorts of directions when you want to" (appendix 1 - Int4.12).

Distinctions which emerge from a repertory grid exercise can be quite subtle and a "revelation" to the subject himself. For example, initially the expert used the term "stop gap" to describe some design solutions. He believed that he used the term "stop gap solution" as a synonym for the term "short term solution". During the exercise it became apparent, however that whilst a "short term solution" was the opposite of a "long term solution" this opposition did not apply to stop gap. After discussion it emerged that the term "stop gap" was used to describe situations where the design proposed would sort out immediate problems but would leave potential problems which would have to be dealt with later on and also to describe quite satisfactory solutions which would solve the immediate problems and which would last until further (more comprehensive) design plans altered the electrical network in such a way as to sweep away the need for the original "stop gap" design.

The repertory grid technique can be a valuable tool for conversation about a domain. The avenues which were explored during the exercise provided valuable insights into the design activity. The exercise helped the knowledge engineer come to understand the more subtle conceptual aspects of the domain by raising issues for further exploration. An example has already been given above (section 3.2.2) of how discussion arising from the grid exercise lead to refinement of the term "obsolescence". Many of the high level terms which appear in the SGNs shown in chapter 7 (in particular the specific context of each design determined by the relevant parts of the existing primary distribution network (figure 7.3) and the abstract context set in terms of guidelines for network design (eventually resulting in figure 7.5)) originated from material which emerged first during the repertory grid exercise and which was followed up either within or as a result of the exercise.

The process of focusing a grid helps to establish a set of constructs which are satisfactory to the expert. If the knowledge engineer uses the constructs to generate a description of an item of experience with which the expert can agree (such as that shown above in figures 6.6 and 6.7) then the outcome of the exercise can be validated to some extent (cf. methodological issues described in section 4 of this chapter).

3.3 Grounded Theory

From the perspective of grounded theory (Strauss, 1990) the qualitative researcher does not begin with a theory about a phenomenon and then look for evidence to prove it, rather the theory

"is discovered, developed, and provisionally verified through systematic data collection and analysis of data pertaining to the phenomenon ... one begins with an area of study and what is relevant to that area is allowed to emerge" (Strauss, op.cit., p.23).

3.3.1 What Grounded Theory Is

The grounded theory approach is a qualitative research method which uses a systematic set of procedures to develop a theory about some phenomenon. The theory is inductively derived from the data. A sub-set of grounded theory procedures can be used to analyze a research question or to develop concepts applicable to it. It was for this latter sort of more limited purpose that procedures from grounded theory were applied to interpret the data collected during the case study. A grounded theory study gets underway from a statement of a research question that identifies the phenomenon to be studied e.g. How do designers justify their design proposals? Questions raised during investigation are supposed to be oriented towards actions and processes. The original research question gets the investigation underway and can be used to re-focus if the researcher loses his way in the data. Coding in grounded theory is the process of analyzing the data and it is coding which forms the core of the theory. Grounded theory is increasingly of interest to information system designers who are concerned to take account of qualitative data (e.g. Brown, 1992). Pidgeon (Pidgeon, 1991) describes in some detail the application of grounded theory for conceptual analysis in knowledge elicitation specifically.

Here are described only those procedures of grounded theory which are applicable to data interpretation leaving aside those for generating a formal theory. The common basis for all grounded theory procedures is as follows : data analysis is a process of interpretation - concepts, hypotheses and so on are constructed *from* the data not found *in* it; the asking of questions is central to all analysis; and the procedures associated with grounded theory should be used flexibly by adapting them appropriately to situations rather than rigidly adhering to them. The three procedures relevant here are open coding, enhancing insight, and axial coding.

Open coding is concerned with breaking down, examining, comparing, conceptualizing and categorizing data mainly by making comparisons and through asking questions (although there are various subsidiary techniques from which to select). Concepts are tentatively labelled and grouped into provisional categories. Categories are developed by beginning to list properties associated with them. Properties are given dimensions. Coding can be started by breaking the data into small units for analysis. For example interview transcripts can be divided into phrases, sentences or paragraphs which contain a point or idea. The analyst proceeds by asking (say), "What is the major idea brought out in this paragraph?". In order to give meaning to the data it is believed that it is important to interleave data collection with analysis. In a situation in which most of the data comes from interviews this means that analysis of the data gives rise to further questions which need to be answered. Insight is enhanced by asking

questions - of the data initially - but also in the interviews when necessary. Strauss and his colleagues offer this advice to qualitative researchers on using interactions to test the validity of possible meanings,

> "All the answers to the questions raised may not be in the actual data that are before you, but there is no need for concern on that account. It is up to you to take the questions to the next interviews and analytic sessions to look for the answers" (Strauss, op.cit., p.80).

Several techniques are associated with enhancement of insight. These are "tricks" which can be used by the analyst to try to break away from his or her own assumptions, pre-conceptions and perspectives. These techniques were not used to any great extent in the case study but they were borne in mind, for example one which gives clues to danger of taking things for granted warns about the appearance of terms such as "never", "always", "everyone knows that" and equivalent expressions - terms like these were always treated carefully.

Axial coding is geared towards discovering and relating categories using the same procedures as those for open coding. Open coding can be thought of as an analytical phase, a decomposing phase, whereas axial coding is concerned with synthesis. Axial coding results in data being put together in new ways by making connections between categories, i.e. with the establishment of relationships among them.

Grounded theory offers a paradigm model for relating data which consists of linking sub-categories to categories in a set of relationships which represent causal conditions, the phenomena, the context, intervening conditions, action strategies and consequences. Open coding, that is the creation and development of categories in terms of properties and their dimensions, is intended to proceed in parallel with axial coding. Any relationships proposed have to be grounded, that is repeatedly supported, in the data. Each step in grounded theory is tested by constantly comparing the emerging structure against the data. What cannot be found in the data determines the boundaries or limitations of the study (Strauss, op.cit., p.112). In short, the aim in grounded theory is not to make generalizations but to specify the conditions under which the studied phenomena exist (op.cit., p.191).

3.3.2 Extent To Which Grounded Theory Was Used

It would be most appropriate to say that the manner in which interviews with the designer were conducted and the way data analysis was interleaved with the interview series did not run counter to the position on qualitative data analysis embodied in grounded theory. In a sense the systemic grammar networks represent a qualitative theory grounded in the data from which they are constructed. The view which underlies the approach taken in this case study is that knowledge elicitation is a process of *generating* a valid description of expertise with a view to constructing a

relevant model that adequately reflects an expert's competence in a context for a purpose (Galal, 1993). It is self-evident, therefore, that the model of competence must be grounded, in the "grounded theory" sense, in the data which is presented.

The techniques of grounded theory which were applied were confined to those which help to organize and structure the data i.e. some methods for open coding (preliminary data analysis) and for axial coding (preliminary data synthesis). Grounded theory leads to the development of a theory which accounts for the phenomena studied. The concern here, however, is with building an adequate model of some aspects of expertise. The value of grounded theory techniques for this study lay in what it offers for organising (for example cross referencing) large quantities of qualitative data; for exploring the data (by asking questions of what has been gathered so far); and for focusing attention on what further enquiry needs to be made. Grounded theory thus played a part in the case study and influenced both the way it was conducted and its outcome. Knowledge elicitation is essentially a process of gathering and analyzing qualitative data from an expert (Johnson, 1990) but the particular nature of what is elicited and the purposes for which it is elicited makes special demands on tools and techniques. Systemic grammar networks, for the reasons given above (section 3.1.3) were found to be the most appropriate representation form for much of the data and the repertory grid technique played an important part in eliciting data particularly that which was rather abstract (section 3.2.3).

4 Methodological Basis for the Knowledge Elicitation

The knowledge elicitation in this case study was conducted within the experimental framework for conducting a conversation as defined by Pask (Pask, 1975; Pask, 1976). In this section the perspective given by Pask's conversation theory relevant to the case study is presented. The account of the theory which follows is based on the description given by Ogborn and Johnson (Ogborn, 1984). It outlines the conditions for a meaningful conversation to take place and explains what constitutes such a conversation. The form and conduct of the knowledge elicitation process as a whole which was adopted for this case study followed as a consequence of these requirements.

An individual "capable of conversation" is defined as an individual capable of thinking and learning autonomously. According to Pask's theory the minimum requirement for supporting conversation, defined in this way, is that an individual must embody processes at two levels. At each level the processes are self-sustaining in that each level can be thought of as a system which interprets itself and can reproduce itself according to that interpretation. (It is the reproductive aspect that gives the possibility of change.)

The two levels, first order and second order, behave as follows. The first order system acts upon the world and through the feedback it receives it is able to make itself more effective. The second order system acts upon the first order system to change its processes. In its turn the second order system receives feedback from the first order system to enable it to improve its own effectiveness. The minimum conversational unit consists of these two self-reproducing levels. In summary, a conversational entity is self-maintaining, it knows what it is like and it is capable of evolving and changing.

Pask's ideas take the form of a theory which specifies the minimum structure that an individual must sustain to be able to know and learn. As a theory, it accounts for everyday observations about what it means to understand something. Pask's ideas also define an epistemological stance which accords due respect to a "knowing subject" by not raising one participant to a superior status from which he or she infers reasons for the other's actions or responses to questions. Conducting a meaningful conversation in which types of questions which elicit data about the higher level procedures neatly dispenses with the need for any commitment to the belief that we can "see inside someone's head" by showing how understanding can be reached without doing so.

One of the most fundamental and valuable features of the theory is that it identifies the conversational entity as the primary unit of analysis. A conversational entity is a stable, self-reproducing conversation. About such entities one can meaningfully say that they know and understand. A conversational entity may or may not inhabit one human body. From this view, capability is not a property of a person but of a conversation so that it makes sense to say that a person has various capabilities or that a group has certain capabilities. For example, it makes sense to say that a school of architecture knows and creates architecture as a group of architects acting together, besides saying that architecture is known and created in a different sense by people who are individually called architects. From this view knowledge can be passed on and known to be passed on by and among a professional group (e.g. a community of experts in the design of electrical power system primary distribution networks). The theory helps in taking a wider view of knowledge in which knowledge is seen as a shared property in conversations, something sustained and evolved, not a deposit in some medium.

The interest here, in eliciting expertise, lies with conversation between two individuals, each of which is capable of conversation. The arrangements for a meaningful conversation to be able to take place between individuals such as an expert and a knowledge engineer are as follows. The first requirement is for the participants to come to an agreement, made public between them, about the domain which is to be the subject of conversation. Conversation then proceeds with the expert describing to the knowledge engineer what he does i.e. what procedures he uses to act on the domain, these are the lower level procedures. The knowledge engineer devises procedures to do what the expert has described

and then tries them out by seeing if they have the same effect as when the expert uses his domain procedures. Once the knowledge engineer's procedures have the same effect as the expert's procedures the participants are deemed to *share some concepts*. The knowledge engineer knows what the expert knows. For the knowledge engineer to come to an *understanding* of the expert the knowledge engineer must ask why the expert uses the procedures he uses in the way that he uses them. If the knowledge engineer can use the expert's explanations to devise procedures which *produce* procedures which operate on the domain the way the experts do the knowledge engineer is said to understand the expert.

To summarize then, the knowledge engineer shares a concept with an expert if the knowledge engineer can make sense of the expert's explanations of what he does. The knowledge engineer shares an understanding with the expert if he or she can make sense of the expert's explanations of his explanations. It is important to notice that there is no claim that the expert's procedures are similar to those arrived at by the knowledge engineer, only that the procedures have the same effects. It follows from this that there is no claim being made that the knowledge engineer knows what is "going on" in the expert's head.

From the perspective given by Pask's theory it is clearly unsatisfactory to restrict knowledge elicitation practices to observation or to the use of verbalisation protocols (e.g. talk-aloud, Ericsson, 1984) to explain the procedures the expert uses to operate on the domain. Often with methods such as these the knowledge engineer assumes for himself a privileged position vis a vis the resulting data by assuming that he or she can infer from the expert's action (or verbalisation about it) something valid concerning the expert's higher level motives, heuristics or strategies. Pask's theory leads to directly ask the expert to explain his explanation. Thus a knowledge engineer operating from the perspective of conversation theory asks questions which seek data about the higher level processes. Examples of the general form these questions take are - Why do you do it like that?, How do you know that is the right thing to do? - in other words questions which ask why the explanations given by the expert of what he is doing *are* explanations.

An "experiment" devised to accord with conversation theory in which a knowledge engineer is to elicit the knowledge of an expert designer requires the following. Firstly, an agreement between the participants about what is to be talked about - in this case study the knowledge engineer agrees with the designer to converse about how the expert designer comes to propose and to justify primary distribution network designs. Secondly, the "subject", the designer, must agree to answer questions about what he does and why he does them. Thirdly, adequate means must be used to ask about and record the descriptions which are needed at both levels. In this case interview transcripts, systemic grammar networks and a repertory grid were used for these purposes. Fourthly, procedures must be used to check back that the procedures devised by the knowledge engineer work to the satisfaction of the designer. Examples of this activity in the case study are

the use of systemic grammar networks to supply the data for use in teachback (described above in section 3.1.2) and the repertory grid check (described above in section 3.2.2). The practical value of Pask's theory is the conditions it sets to guide in the conduct of conversations which will successfully achieve shared knowledge.

End note to chapter 6:

A Note on the Origins of Systemic Grammar Networks

The work of Ferdinand de Saussure marked the beginning of modern linguistics and underpins it as an independent subject of study. His contribution is recorded in the lecture notes reconstituted from his courses in linguistics which were first published in a collected form in 1916 as "Cours de Linguistiques Générale". Many of the important linguistic distinctions which have formed the subject of study during this century were first made explicit by Saussure. He proposed a series of distinctions which have subsequently been developed into the basis of the structural analysis of language. As well as underpinning linguistics itself, the structuralist methods of Saussure have been appropriated for analysis of diverse areas of human discourse, most notably social anthropology (Levi-Strauss, 1963), epistemology (Foucault, 1972), Marxism (Althusser, 1972), literature and culture (Barthes, 1972, 1972a) and psychoanalysis (Lacan, 1977).

The fundamental distinction in structural linguistics is between language (langue) and speech (parole). Language is constituted from lexical, grammatical and phonological components which, according to Saussure, are laid down in the form of rules of language which are acquired by an individual in childhood. The language exists at a level above that of an individual speaker, it arises as a collective product of the community of speakers. When an individual speaks he can only perform or operate within this language. What an individual utters is speech and his choices are restricted to deciding when to speak and what to speak about. The rules of language determine the limits within which individual choice can be made. Language is defined as all possible uses of language - it is an abstraction. Speech, on the other hand, is the actual way in which everyday concrete utterances are realized from the possibilities presented by the abstract language system. The term "structuralist" comes from the emphasis placed on the structural interrelationships which exist between the language and the speech which it permits. Language is what is both implied and presupposed by every single concrete utterance of speech. "Language is a system of interdependent terms in which the value of each term results solely from the simultaneous presence of the others." (Saussure quoted in Kearney, 1986). Structural linguistics is concerned with the study of language defined in these terms.

One of the main ideas underpinning Saussure's analytical method is the realization that the relationship between a signifier and that which it signifies is arbitrary. Signs do not operate in isolation. A sign signifies to the extent that it differs from other related signs, and so its context in a *system* of signifiers is of prime importance in giving it some meaning. Saussure believed that the differences are all that matter in languages, and that the means by which differences are maintained are irrelevant. Acceptance of this idea has obvious far reaching consequences as to what it is meaningful to to analyze in any domain of discourse and how it can

meaningfully be done.

Two classes of relationships between linguistic elements are particularly relevant for the origins of systemic grammar networks. These are concerned with the contribution to meaning which is given by the serial structure of terms, the syntagmatic relationships, and that given by the relationships between comparable terms which can be chosen at a particular place in a structure, the paradigmatic relationships. (These latter relationships were originally termed associative ones by Saussure but are now more commonly called paradigmatic relationships.) These relationships can be loosely described as follows. The meaning of (say) a phrase is partly given by the differences, relative to one another, between the terms of which it comprises a sequence. This gives the syntagmatic relationship between the terms. Paradigmatic relationships confer meaning through the choices which have been made and derive from the difference between the terms actually used and the alternatives from which they might be chosen. The idea of collocation (habitual association of certain words in a language with one another) and the way in which collocatability contributes to the meaning of words is one example of a kind of syntagmatic relation. Whereas the idea that the meaning of a word is, in part at least, determined by the availability to the speaker of other words which are related (semantically) to the one chosen - not solely as some intrinsic property of the word itself - the notion of paradigmatic relations - can be associated with the field theory of meaning.

One aspect of the development of linguistics in Britain which followed on from the work of Saussure is exemplified by the work of Malinowski (Malinowski, 1935) and of Firth (Firth, 1930, 1937, 1957). This is the work which has, as a central concern, the extra-linguistic factors which affect the meaning of utterances, that is theories of the role of situational context. Malinowski's interests as an anthropologist lay in ethnography and in particular in the study of cultural aspects of language. Firth's main concerns, on the other hand, were ethnomethodological. His interest was in categorizing the actual environments within which utterances occur to provide ways of singling out features relevant to the functioning (the meaning) of the utterances.

> "..the force and cogency of most language behaviour derives from the firm grip it has on the ever-recurrent typical situations in the life of social groups, and the normal social behaviour of the human animals living together in those groups. Speech ... is a network of bonds and obligations." (Firth, 1937)

Firth included all "pragmatics" within his field of study including in it the cultural and physical environment and the range of interpersonal relationships which sustain and are sustained by societies. Firth did not regard language as an expression or communication of inner mental states of an individual - "Speech as noise is only operable socially."

> "By regarding words as acts, events, habits, we limit our inquiry

> to what is objective and observable in the group life of our fellows. A piece of speech, a normal complete act of speech, is a pattern of group behaviour in which two or more persons participate by means of common verbalizations of the common situational context, and of the experiential contexts of the participants... His (an individual speaker's) bodily speech habits are established links with his fellow, and he finds it impossible to draw a boundary round his individuality." (Firth, 1930)

Thus Firth's work was concerned with theories of the context of situation, with the use of methods to analyze language polysystemically in terms of the ways it is used. He has not been alone among linguists in studying the role of situation in affecting the semantics of utterances, parallel and complementary work has been carried out in the U.S.A., for example, although Firth's contemporaries there had other primary concerns which directed their work towards ways of describing languages. Since the objective here is to trace the origins of systemic grammar networks the work of Firth is of most relevance and hence has been singled out for attention. It was one of Firth's students, Halliday, who developed Firth's ideas into a linguistic theory (sometimes, owing to its origins, termed neo-Firthian linguistics).

Halliday developed Firth's ideas for interpreting language in terms of its place (function) in the social process. His ideas concern the ways in which language reflects and forms social structures. Halliday is not concerned with what speakers of language know or what is going on in their heads but with what they can say as a realization of an abstract "meaning potential" concerned with what they can mean. Language, in these terms, is seen as a system of meaning potential (Halliday, 1978, p.39). Halliday's systemic grammar is "a theory of linguistic analysis in which languages are described as sets of options (systems) that allow speakers to frame and interpret utterances in response to all the situational requirements as they apprehend them or assume them" (Robins, 1980). So (to repeat part of the introduction to systemic grammar networks given in section 3.1 of this chapter) in Halliday's terms, in order to make sense of what the speaker actually says, "we have to interpret it against the background of what he 'can say' ... it is the actual seen against the background of the potential" (op.cit., p.40). A systemic grammar network represents the potential, the network of options and the inter-relations between them.

CHAPTER 7

Analysis and Interpretation of Data

"insects have their own point of view about civilization a man thinks he amounts to a great deal but to a flea or a mosquite a human being is merely something to eat"

Don Marquis (certain maxims of archy)

The analysis and interpretation of the case study material is presented in this chapter in two forms. The first consists of descriptions of the design process and its outcome given by way of three specific examples of design projects. The second consists of a systemic grammar network (SGN) representation of the design situation. The first form of description is intended to convey the nature of the design activity, the form of the second to represent that which is relevant to competent performance. Both forms of description have been constructed on the basis of the knowledge elicitation carried out using the tools and techniques described in chapter 6.

1 Introduction

During knowledge elicitation discussion with the designer always preceded study of the planning proposal documents. The documents referred to in section 2.2 of chapter 6 have played an important but secondary role. Thus, the primary source resulting in the analysis which follows was the designer. The numbers in brackets of the form Intn:m which appear in this chapter refer, as in chapter 6, to the interview session number (n) and the sections (m) of the transcripts or session notes to which they relate. Where supplementary material has been inserted from the planning report for the design proposal being discussed this is indicated by "PR" immediately following in brackets.

1.1 Examples of Design Projects

In the examples presented below (sections 2, 3 and 4), the design alternatives generated and evaluated by the designer through viewing his design task in different ways - his problem framing - are described. The descriptions have been *constructed* on the basis of the elicited data. No psychological claims are being made.

The interpretation of the data from the knowledge elicitation sessions makes use of the designer's references to a number of design alternatives for each project. These do not correspond on a one to one basis with the

alternatives laid out in the designer's proposal document. For example, in conversation, the designer often made reference to minor variations of design alternatives and to alternatives which were subsumed or discarded for the formal planning proposal. The interpretation is intended to represent a plausible account of the design process, the proposal in document form, on the other hand, is the design product i.e. a design recommendation and support for it.

In the design proposal which is the product of the designer's investigations and deliberations, a fairly formal presentation structure is observed. Usually the proposal document consists of a recommendation and the case for it. The case is made through descriptions of a number of alternatives and discussion of the salient technical and financial considerations which the designer feels should be brought to the attention of the approving authority. In other words, the document makes a case for the recommended action. The three examples of design projects described below in sections 2, 3 and 4 have been selected to illustrate important aspects of the nature of the designer's task. Each example is described by three components.

The first component is a constructed account of the designing shown in two parallel columns. The right hand column (in plain text) gives an account of the activity. It presents the designing in the form of a comprehensible account which links the facts in a plausible way. It consists of an outline of each alternative considered and evaluated and introduces facts from the situation as they become relevant to the decisions or as they are brought to the designer's attention by them. The text in italics, mainly in the left hand column, gives a commentary to the right hand side which expresses what happens in terms of sequences of problem setting (or framing), problem solving and solution evaluation. Each of these three aspects of designing affects the other two in a continuous evolution, each constantly leads the designer to revise his understanding of the situation.

The second component of each description is an account of how the description of the designing which forms the first component (in particular the design alternatives described) relates to the alternatives that feature in the design recommendation (the planning proposal document). Alternatives mentioned in the document appear either as the recommended one, or are described to support the one recommended.

The third component of each description presents the designer's appreciation of the outcome of the design process; this is usually the design recommendation. It shows how the designer characterises the design solution finally approved. This component is based on the constructs elicited from the designer during the repertory grid exercise (described in chapter 6 section 3.2.2). The design projects which form the first two examples (sections 2 and 3 below) were two of the six elements used for the repertory grid exercise, the accounts given here were therefore obtained directly during that exercise. The project which is the subject of the third

example (section 4 below) was not one of the original repertory grid elements. A description of the recommendations for this project was constructed by the interviewer and used to check back with the designer that negotiation of constructs had been successful. (This is referred to in chapter 6 section 3.2.2 and the constructed description itself is given as figure 6.6).

To summarize then, the three components of the descriptions of each example are : an account of the designing, a description of how the alternatives from designing relate to the design proposal, and an account of the designer's view of the recommendation or outcome of the design project.

Following the descriptions of each example (sections 2.1, 3.1, and 4.1) there is a discussion of some of the aspects of designing which are salient. Each discussion section (sections 2.2, 3.2 and 4.2) focuses on different aspects of designing. Each aspect is discussed by reference to the example with which it is first associated. However, the aspects of designing discussed are not peculiar to the example used to illustrate the discussion. The reader is invited to re-read the descriptions of each project with different perspectives after reading the discussion sections for all three projects. The discussion sections attempt to introduce different viewpoints from which to understand the designing described in all three examples. Thus, the issues discussed in sections 2.2, 3.2 and 4.2 can be used to inform a number of readings of the project descriptions.

The first example, concerning diversionary work at Swedenill[1], has been selected to introduce the study by using a comparatively straight-forward design situation. Although it lies at the least complex end of the spectrum of design situations with which the designer is presented, it is nevertheless still representative of the sort of challenge with which he is routinely faced.

The second example, concerning network reinforcement at Breedpace Lane, has been selected for the potential it offers to explore what needs to be defended in a design proposal in contrast to what can remain unstated through resort to the mutual understanding and shared prejudices among the designers and those who make judgements, to approve or reject, the designer's planning proposals. Although also noticeable in the first example, this aspect of design is more prominent in this second example.

The third example, concerning the transformers at Branse sub-station, has been selected because the design situation offers scope for a rich variety of problem framing alternatives. However, it is, at the same time, relatively well-bounded and self-contained because of Branse sub-station's (fairly unusual) geographical and electrical isolation.

---

[1]The names of the design projects used as examples have been changed for reasons of confidentiality.

1.2 Representation of the Design Situation

The second form of analysis is presented in the form of a systemic grammar network. This was constructed to represent a view of the data elicited. The SGN defines a grammar which captures the representable aspects of the design situation relevant for competent designing. The way in which the SGN was constructed and the role it played in prompting, documenting and refining the knowledge elicitation has been described in detail in chapter 6 (section 3.1). The outcome of this work is presented in section 5 of this chapter. Section 5.1 gives an overview of the network. Section 5.2 presents the main aspects of the network in more detail and section 5.3 shows how parts of specific design situations can be seen as paths through the network.

2 Example 1 - the Swedenill Diversion

This example represents one of the simplest design situations that is likely to arise for the attention of the designer. The cause for concern is clearly identifiable and well-bounded both in geographical terms and in terms of its effect on the electrical network in which it has occurred.

2.1 Description

2.1.1 Constructed account of designing

Circumstances initiating design	A particular 66 kV cable route consisting of four cables supplying a 66/11 kV sub-station with a load of 40 MW has become a problem. The number of faults occurring on the cable route is abnormally high* (Int1.2) and increasing (PR). (*All plant, equipment and cable failures are recorded on incident reports (Int1.2, Int1.4) and reliability indices are maintained for an extensive variety of components, hence "abnormally high" is quantifiable.)
Clarification of the circumstances	Analysis of the faults shows that they tend to occur on two of the cables slightly more often than on the other two (Int1.17) and that the faults actually occur in a particular 800 metre stretch of the 6.5 kilometre route (Int1.9). This section of the cable route is identified as corresponding to a piece of diversionary work carried out about 25 years previously.

These circumstances can be viewed, i.e. the problem can be framed , as a need to supply the load from other parts of the network (alternative 1)

By concentrating attention on the need to supply the 40 MW load by securer means, the designer considers whether the load can be transferred elsewhere (i.e. be supplied from other sub-stations) (Int1.22) which would permit the sub-station fed from the faulty cables to be shut down, that is, taken out of service.

Viewed in this way certain factors become relevant , namely particular properties of the existing , surrounding electrical network and general knowledge about matters likely to affect these properties

The surrounding area is well loaded, with no reasonably adjacent large block of spare capacity (Int1.6, Int1.7). Discussions are underway with various (property and land) developers whose proposals, while tentative, may result in additional loads within the area supplied by the sub-station (PR).

Evaluation of alternative 1 is an argument directed by this view of the problem, and takes into account the factors brought into focus by it

The existing load of 40 MW, normal load growth, and some of the development proposals can be accommodated by the existing sub-station. However, should all the development proposals come to fruition, reinforcement of the sub-station will be necessary (PR).

Further relevant properties of the surrounding (electrical network) context are brought to bear, these impose practical security constraints

Moving the load on to spare capacity in the surrounding area is not practical (Int1.22).

Design commitments are brought to bear on the argument

In addition, moving the load would be contrary to good design practice which aims to reduce system losses (losses of electrical energy). It would run counter to one of the ways of minimising losses which is to establish supplies of energy (sub-stations) close to the centres of the load they serve[1] (Int1.32, Int1.36, Int 1.39, Int8.13).

As a consequence of the argument another way of viewing the problem suggests itself namely as a need to design an improved supply to the existing sub-station, this leads to a series of associated alternatives

The simplest alternative (alternative 2)

Replace faulty pieces of cables with similar ones (Int1.14). (Similar with respect to operating voltage and current carrying capacity.)

Viewed in this way certain properties of the surrounding (network) context become relevant..

The rest of the cable on this route is between 50 and 60 years old. The plant the cable supplies is between 20 and 30 years old.

.. with some general knowledge based on experience ..

There are no particular problems associated with the kind of plant installed (Int1.12).

..and a heuristic

The normal life* of cables is estimated to be about 80 years and for plant (switchgear and transformers) it is about 40 years (1.12). (* This is a rule of thumb; equivalent ones are at work throughout the electricity supply industry, values vary depending on local conditions and types of plant. In practice, network planners and operators actually work with something much more subtle than this neat heuristic (Int1.12, Int1.13, Int1.14).)

Evaluation of alternative 2 is an argument directed by this view of the problem, taking into account the relevant factors brought into focus

In just over ten years time plant will need to be replaced at the sub-station supplied by the cables.

Design commitments are brought to bear on the argument

At that time the opportunity will be taken to re-design the sub-station reducing the amount of equipment installed. The current network development strategy which the designers work within achieves a reduction of equipment, among other ways, by removing intermediate points of electrical energy transformation on the network when the opportunity to do so presents itself[2]. The practical consequences of this design practice in this situation is that the four 66 kV transformers will be replaced by two 132 kV transformers (Int1.16, Int1.24).

Evaluation of alternative 2 leads to modification of the problem framing which gave rise to it - from a need to design an improved supply to the existing sub-station to a need to supply the sub-station as it will be configured in the longer term (alternative 3)

Alternative 3

Replace two of the faulty cables (the worst two) with uprateable cables (capable of supplying two 132 kV transformers) leaving the two least-bad cables in service (Int1.17).

This brings to bear other relevant

The surrounding area is well loaded, with

factors from the surrounding context..

no reasonably adjacent large block of spare capacity (Int1.6, Int1.7). There is a sensitive load in this supply area (Int1.18). This fact is offered as supporting argument when considering the consequences of further deterioration of the cables. A sensitive load, in this case a high-profile consumer, would not merit special provision per se, but can be, and in this case is, introduced to support the case for a particular alternative.

.. and further reference to the initial circumstances

There has been at least one occasion of overlapping outages on this circuit (when a second cable has faulted during the repair operation on another (the first fault)) (Int1.25).

Evaluation of alternative 3 is an argument directed by this view of the problem taking into account the relevant factors brought into focus

The two remaining poor cables may deteriorate further in the next few years (Int1.17). Moving the load onto the surrounding network is difficult (Int1.20), large "chunks" cannot be readily accommodated (Int1.20).

Hard constraints are met by this alternative but a commitment to the "spirit" of these constraints is brought to bear on the evaluation

This amounts to a problem with meeting the standards for security of supply (i.e. the security constraints); specifically, load transferring activity would exceed the time limit dictated by the security of supply standard (Int1.21) in the event of a second outage. Consequently, the possibility of transferring load post-fault quickly enough is not considered feasible (Int2.7), although strictly speaking the situation is acceptable according to the security of supply standard.

The arguments brought into focus by evaluation of alternative 3 lead to the proposal of an alternative (alternative 4) that will overcome the drawbacks of alternative 3 whilst still meeting the need from which alternative 3 arose

Alternative 4

Replace two of the faulty cables with uprateable ones and replace the other two by installing two "cheap" cables which can be discarded when the sub-station configuration is altered in the future when new plant is installed (Int1.18, Int1.19).

1 The strategy of placing points of supply close to load centres relates to the desire to reduce system losses and also to avoid voltage regulation problems (Int8.14, Int8.15).

The relationship between these design commitments, and their relationship to others are shown in the systemic grammar network in section 5.2.4 of this chapter.

2 The ways in which commitments to reduce the amount of equipment installed is achieved (Int8.13, Int8.17), and its relationship to other design commitments is shown in the systemic grammar network in section 5.2.4 of this chapter.

2.1.2 Linking the designing to the design recommendation

The first alternative presented in the account above (section 2.1.1) does not appear in the planning proposal. It is ruled out, the designer says, "it would have been thought of and dismissed probably without realizing it because of the loadings involved" (Int1.22). Effectively, given the context of a densely built-up urban area, the idea of removing a sub-station currently supplying 40 MW and supplying the load local to it from elsewhere is unlikely to be a good idea unless there is some compelling reason to consider it. The fact that it is not even mentioned in the planning proposal indicates that the "automatic" ruling out of this alternative is a generally acceptable assumption (Int2.6). The issue of what is stated and unstated but mutually understood is discussed in relation to this example later in section 2.2.3 and for the second example in section 3.2.3.

The remaining three alternatives given in the account above are presented in the planning report. Alternative 3 is ruled out on the basis of engineering considerations which correspond to the evaluation arguments presented above. The designer's decision of what to recommend rests on financial considerations as well as engineering ones. The weight attributed to financial considerations is based on the relative importance being attributed, at a particular time, to the various benefits offered by the alternative designs. The weighting is a "political" matter. Even in this example, one of the simplest situations of its kind, it can clearly be seen that the alternatives are not functionally equivalent to one another, so that a simple choice - of the cheapest - cannot be made straight forwardly. Judgement of what to recommend, and of what will be acceptable, essentially rests on knowing the amount the organization is currently able or prepared to pay for the various benefits offered by the alternatives (Int1.18, Int1.23).

Alternatives 2 and 4 are presented in the planning report and a recommendation is made on the basis of the arguments presented through the alternatives described in the report (Int3.2, Int3.32). This includes a cost comparison between the alternatives based on nett present value calculations. The recommendation reflects the prevailing attitude in the organization towards financial investment in a design which offers future functional benefits which must be paid for in advance.

The cost of the solutions outlined are between one and two million pounds. The designer can be seen to make opportunistic use of the situation i.e. to make improvements to the electrical network as a whole, in

accordance with what he values. It can be seen that here, for example, as part of the recommendation, it is proposed that the new cable is re-routed to avoid a heavily congested transportation route along which the old, faulty cable was routed (Int2.1, Int2.2). This is further justified on the basis that the existing route is believed to be likely to be the subject of road improvements in the next few years (PR). The designer refers to this use of an opportunity as "using a little local knowledge" (Int2.2).

2.1.3 Designer's appreciation of the recommendation

The designer views the final recommendation as essentially one of expediency in that the question of justification of advanced expenditure (Int5.11) i.e. paying now for a future benefit was considered and a decision was made not to spend the extra money (Int3.3, Int5.4). The recommendation offers no longer term benefit (Int5.9). It "gets over an immediate problem" (Int5.13) although potential problems remain since old equipment remains in service (Int4.10) in the form of old cables along the route (Int4.17).

2.2 Discussion

This first example of designing, although one of the simplest and most straight forward of its kind, clearly illustrates Lawson's characterization of designers as being "solution orientated" (Lawson, 1990, p.32). As the designer explores possible problem framings and the solutions which can be associated with them he learns more about the design situation with which he is faced. This is reminiscent of Gero's description of design as an exploration process where what is relevant only manifests itself as the design proceeds and varies with the decisions taken (Gero, 1990, p.29).

2.2.1 The task is a reflective exploration

The example as it is presented above (2.1.1) shows that the design task can be viewed (using Schön's terminology) as a reflective conversation in which as the designer frames different views of the situation and develops ideas about solutions. Within each framing factors from the surrounding context are brought into focus. Argument is stimulated as the situation talks back and design commitments and constraining influences come into play as they become relevant. This unfolding finds echoes with the comments of Fischer that every step made by a designer towards a solution determines a new space of related information (Fischer, 1991, p.192). This sort of description of the design task can be seen to fit equally well for the two examples of projects which follow this one (sections 3.1 and 4.1). Acceptance of the view of designing which this description fits renders unsuitable those representations of the design task which are equivalent to non-adaptive A.I. planning representations. Representation of movement through a task structure which does not capture responsiveness to the data of the design situation as it unfolds (becomes relevant) during exploration

will unnaturally constrain any designer trying to interact with a system based on that representation.

It may be possible to represent many of the aspects of a design situation that a designer may consider during designing. However, the order in which he will consider different aspects, and which ones will even be relevant in a specific case, will be determined by what emerges from the situation during exploration. Knowledge elicitation and analysis may allow construction of representations of some aspects of a design situation which are potentially relevant to designing, but on the basis of the view of the design process given above, procedures that act upon the represented structures must be flexible. They must be directed by the results of exploration, and this must be under the control of the designer.

2.2.2 Design commitments are used to evaluate design alternatives

Design commitments can be seen to play a significant role in the evaluation of potential designs. They can be seen at work not only for evaluating a design alternative but as an active force in suggesting and shaping alternatives. In this example, design commitments contribute to the generation of alternative 3 on the basis of evaluation of alternative 2. Alternative 3 arises from addressing the factors brought into focus by applying design commitments (specifically one relating to reducing the amount of installed equipment) to alternative 2 and finding it wanting in this respect.

A further example of the generative role is observable in the "mutation" of alternative 3 into alternative 4. Alternative 4 addresses the shortcomings of alternative 3 with respect to a commitment to the "spirit of the law" embodied in the requirement to satisfy security constraints. Here we see some subtlety of application of a "hard" constraint. Alternative 3 meets the "letter of the law" as far as the security constraints are concerned. It satisfies the regulatory requirement since it is feasible (electrically) to resupply the requisite load, in the requisite time in the event of a fault occurring. (The security constraints are represented in the form of a systemic grammar network in section 5.2.5 of this chapter.) However, the spirit of the law is also a matter of concern, there is a commitment to design systems such that it is also practically[2] possible to reconfigure the electrical network fast enough to meet the security constraints.

The same commitments (to security constraints) also play an eliminative role in the evaluation of design alternatives. Since security

---

[2] To reconfigure the electrical network engineers must travel between sub-stations to carry out switching operations. Here we are faced with a congested city centre where the time to travel between sites depends on the traffic, the time of day, etc. Viewed strictly, the security constraints do not require consideration of these practicalities. They impose constraints which must be met by the electrical properties of the network (e.g. interconnections, spare load carrying capacity).

constraints must be met (i.e. they are "hard" constraints) it is not surprising to find them playing the role of eliminating an alternative. In this first example it is alternative 1 that is ruled out by not meeting the security constraints.

2.2.3 Explanations "why not" support the design

During the repertory grid exercise the designer explicitly recognised that one role of the design alternatives in a planning proposal is to show that a course of action, or an idea is *not* feasible or *not* recommended. The designer described this role as one of using design alternatives "to rehearse the argument" (Int4.9) to provide support for the design alternative chosen for recommendation. In this example the inclusion of alternative 3 in the planning report fulfils this role. As we have seen from its evaluation, this alternative is not a feasible solution to the design problem which has been framed. Alternative 3 is described so as to support the case for alternative 4 with a view to its comparison (ultimately on a financial basis) with alternative 2.

It is noticed that not every aspect of a design alternative has to be justified. This would be impossible. The norms of the professional group involved in designing and approving the design proposal play a part in what is justified. The designer knows what can be left unsaid and what arguments must be presented. The appearance of alternative 3 in the proposal is implicitly to explain why alternative 4 is what it is (specifically why four cables are needed rather than two). Here we see the role of alternative 3 as one of removing or averting a misunderstanding that might otherwise arise (cf. chapter 5 section 1) among the norm sharing group. The explanation why alternative 3 is not suitable supports the case for alternative 4.

By contrast it is noted that the problem framing discarded with alternative 1, namely the idea of transferring load elsewhere to dispense altogether with the need for the circuit containing the faulty cables, is not mentioned at all in the proposal. Norms are at work here also. As already noted, load transfer under these circumstances would be dismissed immediately (but compare this with the discussion of example 2 (section 3.2.3) for the situation in other circumstances where the context sensitivity of explanations is discussed). The professional group are in accord on this point; it is not mentioned because, to them, its exclusion is taken for granted. There is no potential misunderstanding over its exclusion; so it is not explained.

3 Example 2 - the Breedpace Lane Reinforcement

This example deals with a situation where load growth in a particular area can no longer be handled by the existing network. The designing can be viewed as a fairly straight forward evolution of ideas. There is a sense of

gradual progression, a linear movement towards a design solution that is satisfactory from more and more points of view. However, the monotonic aspect of progression is not always the case, as will be seen in the third example which exhibits greater complexity. First, however, in this example, three different problems are framed which lead to the generation of five design alternatives each of which offers a means of reinforcing the electrical network to a different extent and by different means.

3.1 Description

3.1.1 Constructed account of designing

Circumstances initiating design

An area of the network is becoming overloaded due to general load growth in the locality (Int2.15). The overloading has been exacerbated by the decommissioning of a 132/6.6 kV sub-station two or three years previously - on safety grounds - the equipment being obsolete and unrated (Int2.13, Int2.30). This sub-station had previously fed some of the load in the area.

One way of framing a problem from these circumstances is as a need to supply the load in the area from the network surrounding the overloaded part (alternative 1)

The designer considers whether load supplied by the overloaded network can be accommodated by sub-stations in the surrounding area; he considers transferring the load elsewhere.

Viewed as a load transfer problem, the surrounding electrical network is considered in terms of those of its properties relating to its ability to supply further load

The most obvious candidates are two 132/11 kV sub-stations (which will be referred to as S1 and S2) which already have been subjected to significant load transfers from the decommissioned sub-station. These two sub-stations are regularly operating at twenty to thirty percent overload under normal operating conditions (Int2.15). Other sub-stations in the area are already well-loaded. The overloading causing concern is on an 11 kV secondary distribution network. Much of the surrounding network is operating with a 6.6 kV secondary distribution voltage.

Evaluation of alternative 1 is an argument initially directed by this view of the problem and the surrounding facts rendered relevant as a result of it.

Significant spare capacity in the adjacent network which might be useful for taking up some of the load from the overloaded area is not available. The difference in secondary distribution voltages (11 kV in the overloaded area, and 6.6 kV in the surrounding area)

renders major load transfers impractical.

Consideration of factors from the surrounding context initiated by evaluation of alternative 1 brings further properties into view for consideration

One of the most severely overloaded sub-stations, S1, is physically adjacent to a 132 kV grid site at which there is an unused transformer bay and a spare 132 kV connection (Int2.16).

These factors broaden the horizon of what is relevant, or shift the designer's focus of attention, leading to new opportunities for problem framing

Due to alterations elsewhere on the network there is a spare 15 MVA 132/11 kV transformer and a spare 11 kV switch panel available (Int2.16).

The designer sets his own boundaries as to what is relevant, he can "add" factors to the initial circumstances to allow a new problem to be framed. Here he frames a new problem as a need to increase sub-station load carrying capacity to relieve the overloaded area.

Alternative 2 results from this problem specification

Install the spare 132/11 kV transformer in the vacant enclosure at the grid site and add the spare switch panel to the switchboard at the adjacent overloaded sub-station , S1 (Int2.16), to complete the connection of this 15 MVA of re-inforcement.

Evaluation of alternative 2 is an argument based on this view of the problem which makes use of relevant factors brought into focus by considering it as a potential solution

This is an inexpensive proposition since site, buildings and plant are already available. However, the spare transformer's capacity will only relieve the present overloading at the sub-station to which it is connected (namely S1). It will not assist with the overloading at S2 nor with the more generally widespread overloading problem. Also it will not cope with any further load growth at S1.

Further relevant aspects of the situation are brought into focus...

A small switch room is vacant at S1, and at the grid site previously mentioned there is room to accommodate transformers. At S2 there is no room for expansion - either in terms of space or in terms of increasing the supply to the area at 132 kV.

... including some general knowledge

There are a number of 132/6.6 kV transformers spare as a result of network uprating programmes elsewhere on the network. (Transformers can be uprated from 6.6 kV to 11kV on the secondary side by being re-wound.)

Once the idea of reinforcing the network by adding energy transformation capacity to an existing sub-station has been entertained it is a small step to consider designing a new sub-

station. The third problem framing is therefore in terms of a need to commission a sub-station in the overloaded area. The remaining alternatives (3, 4 and 5) arise from this view of the problem.

Alternative 3

Establish a new sub-station* in the available space (i.e. a switch room at S1 and transformers at the adjacent grid site) to reinforce the overloaded network (Int2.17). Possibly use the spare (and some re-wound) transformers (Int2.18). Use the spare switch panel to establish inter bus-bar connections between the two switch rooms at S1. (*Note that a site can be occupied by one or more sub-stations.)

Evaluation of alternative 3 is an argument based on considering the implications of what it proposes

There is a "default" configuration - a standard sub-station layout - for 132/11 kV substations. There is insufficient space at the proposed site to accommodate this configuration (double bus-bars with four transformers (Int2.7)) if the regulatory escape passage way clearances are to be complied with (Int2.21, Int2.22).

Evaluation rules out alternate 3 as it stands; a heuristic is relevant, a hard constraint (security constraint) is met straight-forwardly by the standard sub-station configuration, deviation from the routine would require explicit justification in terms of how the security constraint will be met

The security of supply standard is usually readily met by a 132/11kV standard sub-station configuration (Int2.8).

Alternative 4, based on the same view of the problem as alternative 3 follows naturally from overcoming that which eliminates alternative 3.

Alternative 4

Commission a new sub-station with a non-standard configuration, namely, single busbars and two transformers (Int2.10, Int2.18, Int2.22). The spare 15 MVA 132/11 kV transformer and another of 30 MVA capacity (possibly a re-wound 132/6.6 KV spare one) will be used with the spare switch panel. The panel will establish inter bus-bar connections between the two switch rooms at S1 (as in the proposal of alternative 3).

Evaluation of alternative 4 includes consideration of how the proposal

Since a non-standard configuration is proposed, the designer must demonstrate

accords with commitments to the security standards (hard constraints)

this design alternative's compliance with the standards for security of supply (Int2.19), Int2.23, Int2.25). The capacity it is proposed to install will provide relief to the current overloading on both sub-stations S1 and S2.

Consideration of how hard constraints are met exposes further qualities of the design proposed

Calculations associated with security of supply issues show that the amount of load at risk in five years time, based on current load projections for the area, will be high (1200 MW hours) (PR).

Remaining with the same view of the problem, the poorer qualities of alternative 4 suggest a modification, giving a solution, alternative 5, which retains alternative 4's advantages but improves performance in other areas.

Alternative 5

Establish a new sub-station with a non-standard configuration in the available space (i.e. a switch room at S1 and transformers at the adjacent grid site) to reinforce the overloaded network. Install two transformers each of capacity 30 MVA. (Possibly use two spare 132/6.6 kV transformers with re-wound windings.) Use the spare switch panel to establish inter bus-bar connections between the two switch rooms at S1.

Evaluation of alternative 5 shows that the drawbacks of alternatives 3 and 4 are overcome with this design

The capacity it is proposed to install will provide relief to the current overloading on both sub-stations S1 and S2. Consideration of adherence to the security of supply standards yields the fact that the amount of load at risk in five years time based on current load projections in the area will be reduced over that at risk with alternative 4 (to about 140 MW hours) (PR).

3.1.2 Linking the designing to the design recommendation

The alternatives presented in the analysis, at least from alternative 2 onwards, display a steady progression towards a design solution which is acceptable from an increasing number of aspects. The first alternative, that of providing relief by transferring load, is mentioned briefly, almost in passing, in the planning proposal, it is not given the status of a viable alternative but serves firstly to show that it has been considered and secondly as a reminder to inform the reader (in particular the one who will approve the design) about an important relevant aspect of the electrical network in the affected area. This is the fact that there are two secondary

operational distribution voltages (11 kV and 6.6 kV), a circumstance which has a significant bearing on what is practical.

It can be argued that alternative 3 as presented above is not a viable alternative as it is stated, since the establishment of a standard sub-station consisting of four transformers with a double bus-bar switch gear configuration cannot be considered as a physical possibility in the space available. However it can be seen that the main themes of the problem framing are load transfer, sub-station extension, and (new) sub-station establishment. Alternative 3 allows the third theme to be established, with alternatives 4 and 5 representing refinements into practical, feasible proposals of the problem framing which led initially to alternative 3.

In the planning document the alternatives presented as such are equivalent to alternatives 2, 4 and 5 in the account given above. It is alternative 5 which is recommended as the design for approval. This one is shown to be more attractive than alternative 4 primarily on the basis of a financial comparison, a position which is further supported by the calculation of the value of the MW hours which are at risk. The appearance of alternative 2 signals that an extension of the existing sub-station, S1, is not a satisfactory approach to overcoming the main overloading. Alternative 2 supports the case for alternative 5 (the recommendation) since it shows that sub-station extension is inadequate and therefore that a new sub-station is required. It also exercises the space issue and by doing so it prefigures the non-standard sub-station configuration which is recommended.

The report discusses the idea of using existing spare transformers with re-wound secondary windings for both alternatives 4 and 5 but finally recommends the purchase of new transformers on the basis of a straight financial comparison (comparing the capital expenditure with the cost of losses each case in a nett present value calculation).

3.1.3 Designer's appreciation of the recommendation

The recommendation completely resolves the initial concerns which prompted design activity (Int4.5). There was some urgency to get something done about the overloading. This affected the decision about what to recommend (Int4.19). In some senses it is a short term solution in that it is anticipated that the particular part of the network being reinforced as a result of the design proposal will be affected by plans to reinforce the network more generally and comprehensively (plans which affect the whole of the central urban area) (Int4.10, Int4.22) . However, the design proposed does not conflict with the future development plans, it offers a tidy solution to the problem presented and allows scope for expansion. The design proposed "does not run contrary to" (Int4.13) the wider development plans; in particular it fits in with the view taken on the way in which the security constraints are to be interpreted (Int4.14) in future development.

The problem concerns local load growth, there is no plant or equipment failure or obsolescence involved. The existing network components in the

area to be reinforced (essentially the installation at sub-station S1) have an expected life-time compatible with the expected useful life-time of the design being recommended (Int6.4).

3.2 Discussion

3.2.1 Task movement is influenced by evaluation

The evaluation of alternatives which takes place during design can be seen as testing the "fit" of a design alternative to the design problem as it is currently framed. Alexander's idea of misfit (Alexander, 1964, see chapter 4 section 4.3) is as a relationship between what is designed (a design alternative) and what is known to be required. This idea must be grasped as a relationship which embraces much more than might immediately be obvious.

Firstly, what is "known to be required" refers not only to the problem or need as framed by the designer in explicit terms but also refers to the demands which are implicit from the norms and practices of the professionals involved in the design environment. Some of the more concrete aspects of these demands are strategies and development plans e.g. compliance with perceived future needs or changes likely to affect the design. The more abstract influences will be determined by norms and agreements among the professional group about their aims and objectives - essentially about what they expect, or desire the designed environment as a whole, to be. These are their commitments. A design alternative that runs counter to design commitments will be sensed as misfitting just as much from these less explicitly stated "requirements" as from the design "requirements" in the more restricted sense of the phrase i.e. those arising from the initial circumstances giving rise to the design activity in the first place.

Secondly, it is important to accept that misfit is a relationship between a design alternative and the problem framed by the designer. Both parties to the relationship, namely that which is designed and that for which it is designed, change as the designing proceeds. As we have seen, the designer sets his own boundaries, he defines both the problem and the solution to fit it. As he goes about this task misfits claim his attention during evaluation of design alternatives. Misfits are often unsatisfactory aspects of the solution proposed. In the example here unsatisfactory aspects of design solutions are seen being brought to the designer's attention during evaluation. In this sense the idea "that a design is unsatisfactory is useful information which helps the designer to proceed" (chapter 4 section 4.3) is illustrated to some extent. However, as demonstrated in the next example (section 4 below) positive qualities of a design alternative that come to light during evaluation may also lead the designer to redefine his notion of what a good fit will include.

Generally therefore, the critical evaluation of alternatives influences what the designer attends to, and therefore it influences what he does next

(the task movement). Detection of misfits prompts him to consider how they can be overcome by reframing the problem and / or by changing the solution. In the Breedpace Lane reinforcement example the account of designing can readily be seen in these terms.

3.2.2 The unexpected prompts justification

A design which includes a proposal to install a standard sub-station configuration needs no justification of its acceptability on the grounds of conforming to the constraints of the security of supply standard. In this example the inability to propose a standard sub-station configuration signals a breakdown of what is expected; something "surprising" is proposed. This demands justification. The non-standard configuration is justified in the planning proposal by paying detailed attention to how the sub-station configuration proposed will comply with the security standard. This is effected by showing how each possible failure of a single transformer can be accommodated by switching operations on the network to re-secure supplies, i.e. the running arrangements in the event of failure of any one transformer from S1, S2 or the new sub-station are explicitly rehearsed in the design proposal (PR, Int4.15).

In this example we can see that lack of space in which to place a standard configuration sub-station leads the designer to pay attention to the standard configuration. In a sense he ceases to take it for granted, he considers why it is the standard configuration and how it meets the security constraints. This consideration leads him to design a satisfactory, but novel alternative. When the standard configuration fits, it is used unreflectively by the designer and accepted equally without question by the professional group and in particular by those who must approve a design proposal. In Heidegger's terms (chapter 5 section 2.2), it can be said that the standard sub-station configuration is ready-to-hand, available for use without inspection. In the situation here, however, the standard configuration becomes present-to-hand, it is inspected, it becomes an object for study. The designer reflects on it; the rationale for it is explored. From this reflection the designer is able to produce a new proposal which satisfies the same constraints which normally render the standard configuration so convenient a solution.

3.2.3 Explanations are context sensitive

In the first example, the Swedenill Diversion, the possibility of proposing a load transfer scheme was not mentioned in the design proposal (section 2.2.3) as a possible alternative. The "explanation" such a description would have furnished was unnecessary in that context for the reasons already given. In this example, the idea of designing a load transfer scheme is mentioned in the design proposal. Its role is to pre-empt a question which the designer sees would otherwise arise. The context here is, firstly, that a load transfer programme has recently been effected in the area (see the description of the initial circumstances given in 3.2.1 above). Secondly, the load transfer option might be believed to be a reasonable design option to support (by covering the abnormal operation condition - i.e. a fault) an

alternative which would extend one of the sub-stations (S1). A reader of the report is disabused of this notion by being "reminded" about the voltages of the secondary networks in the area which render the load transfer idea unacceptable as candidate. The explanatory text in the proposal document is as follows "Three major options have been considered as a means of providing relief to the sub-stations at S2 and S1 with the minimum delay. These options do not include that of transferring load to adjacent sub-stations as the adjacent sub-stations have either insufficient spare capacity available or an operational secondary voltage of 6.6 kV" (PR). The designer judges when explanations are needed based on the audience of his proposals and his understanding of what is relevant from the design situation. He puts it thus, "there are obviously assumptions built in, that the people the proposal is going to be read by know the system, although not as intimately as oneself" (paraphrased from Int3.31).

An explanation is a communication and successful communication requires, among other things, some shared agreement about the context and, for a written communication, a notion of the audience. The designer made several references to his conscious consideration of the audience, usually focussing on those whose task is to approve the design proposals (some examples of these appear in Int1.34, Int6.30, Int6.45, Int7.8). "If you put a group of engineers together ... and if they say 'well, have you considered this?' and if you haven't considered it ... they'll turn your work back to you ... so it's a little bit of a guessing game ... which options are they going to give serious consideration to and which ones are ridiculous ... you try to encapsulate those (i.e. the former)" (Int 3.40). The designer consciously acknowledges the tactic of pre-empting, in the design proposal, the questions that might otherwise be asked. He describes it thus, "... over time a body of knowledge builds up about what sort of questions will be put by the people in the organization who approve the proposals ... the engineer becomes accustomed to these 'ways of thinking' and bears them in mind when preparing proposals" (notes of interview 9).

To describe explanations as context sensitive might almost be misleading for the notion of a context-free explanation is nonsensical. It is part of the designer's practical knowledge to know when explanations are needed and to understand what will constitute adequate ones. In other words, the designer knows both under what circumstances a misunderstanding might occur and what the nature of it would be should it occur.

4 Example 3 - the Branse Transformers

This example concerns a situation where the transformers at a geographically and electrically fairly isolated sub-station have become leaky and unreliable due to old age. It is a situation which offers a rich variety of ways of framing a problem, depending on what is considered to be within the designer's "brief", it offers a correspondingly radically varied selection of solutions.

4.1 Description

4.1.1 Constructed account of designing

Circumstances initiating design

A sub-station, Branse, supplying a geographically and electrically isolated area (Int6.8) provides energy via four 15 MVA 66/6.6 kV transformers. These transformers have reached the end of their natural life. They were installed sixty years previously (Int6.9) and now are leaking oil to an unacceptable degree, the tap changers are obsolete and break down frequently (Int6.10), and revenue costs associated with the transformers is about seven times the cost of maintaining other equivalent plant (Int6.12).

These circumstances can be viewed, that is, a problem can be framed as a need to provide transformation facilities to 6.6 kV at Branse. This suggests a number of alternative design solutions, the most straight forward gives alternative 1

Taking the simplest approach, the designer considers carrying out a straight replacement at Branse, replacing the old transformers with newer ones of the same capacity and voltages (Int6.13).

Viewing the circumstances this way, some general knowledge becomes relevant

There is a plentiful stock of second hand transformers which have been released from duty as a result of system uprating* projects elsewhere (Int6.13). (* That is uprating of secondary distribution networks from 6.6 kV to 11 kV.)

Evaluation of alternative 1 considers its impact thus bringing into focus further aspects of the Branse installation and the supply to it

The old transformers are of very low impedance, as a result reactors have been installed at Branse to keep the fault level down to an acceptable level (Int6.11). Newer transformers (and thus any replacements) are of higher impedance so coupling them with reactors is unnecessary (Int6.14). The energy supply to the transformers is via four cables; two of which are oil filled cables in good condition, two of which are 40 year old gas filled cables of an age where leakage is expected to be a problem* (Int6.12). Some poor performance has already been noticed on the gas cables' circuit. The load supplied by the Branse transformers is about 30 MVA; no load growth is expected in the next few years, in fact a slightly negative

load growth is fairly likely (Int 6.13). (*Heuristic - after about 30 years of service gas filled cables start to leak; life expectancy is about 40 years.)

Evaluation of alternative 1, paying particular attention to its negative aspects suggests a second alternative (alternative 2)

The designer considers replacing the four transformers with four recovered from elsewhere (as with alternative 1) but connected by double banking* onto the two oil filled cables (Int6.14). (* Connecting two transformers on to each cable.)

Evaluation of alternative 2 brings properties of the surrounding network into relevance and commitments to security standards are brought to bear

The oil filled cables have sufficient current carrying capacity to supply two 15 MVA transformers each (Int6.14). The security constraints are met[1] by the configuration proposed since should one oil filled cable fault, the load at Branse could be held by the two remaining transformers supplied on the other cable.

Evaluation of alternative 1 also draws attention to the load a Branse, this suggests a further alternative very similar to alternative 1, alternative 3

Since the load at Branse is (only) 30 MVA, the designer considers replacing the four 15 MVA transformers with three (Int6.24). These would be supplied from the two oil filled cables and one of the gas filled ones. Security constraints demand that each transformer be supplied on a separate cable (Int6.27).

Evaluation of alternative 3 shows comparison being made between the (competing) alternative solutions

This solution would leave one gas filled cable in service and in this respect it compares unfavourably with alternative 2 (Int6.27). The cost of the extra transformer required by alternative 2 would therefore be weighed against the benefit of getting rid of the gas filled cables. The cost of the transformer is likely to be low since it will be one recovered from elsewhere on the electrical network. In a sense alternative 3 effectively eliminates alternative 1 since it has two clear advantages over it, namely one less transformer, one less poor gas filled cable left in service.

By including further circumstances in his deliberations, the designer is able to frame the problem in another way. This leads directly to two further alternative solutions.

Further circumstances which can be included as a starting point for

The switchboard at Branse will need to be replaced in about 10 years time at the end of

designing

its normal service life (Int6.15). The Branse sub-station occupies a large site. (The old transformer and reactor pairs are bulky plant.) The site is in an amenity area; the site is a valuable one (for redevelopment) (Int6.16). The sub-station effectively occupies two buildings one of about twice the area of the other (PR).

These factors, combined with the initial circumstances allow the designer to frame the problem differently, he attends to the idea of releasing part of the site for redevelopment. This means he sees the problem as a need to redesign the sub-station at Branse and having it continue to supply the area it currently supplies(Int6.32-6.35).

Alternative 4

The designer considers installing new CER (combined emergency rating) 12/24* MVA transformers to meet the load at Branse; the load will be supplied through a new switchboard (Int6.17). (*Normally supplying up to 12 MVA, in emergency (during an outage) capable of carrying 24 MVA (Int2.7).)

Consideration of commitments to the security constraints -already brought into focus for evaluating alternatives 2 and 3 -lead to completion of this solution

An autotransformer (11/6.6 kV) will be installed to provide infeed from the surrounding secondary network in the event of a fault (Int6.18). (Briefly, should one 12/24 MVA transformer become unavailable due to some fault, the other, operating at 24 MVA loading will be unable to meet the load at Branse (30 MVA), hence the infeed at 11 kV to make up the shortfall (Int6.17).)

Evaluation of alternative 4 makes reference to the broader circumstances which are brought to bear in the problem framing

This proposal will release the larger building at Branse and the land it occupies for resale (Int6.17). Although the cost incurred for replacing the switchboard is advanced by 10 years, this will be more than offset by the resale value of the land released by the proposal (Int6.21). It , like alternative 2, will dispense with the poorly performing gas filled cables.

Design commitments are called into play in evaluation of alternative 4

It will be thought unusual to install a new sub-station transforming to 6.6 kV. This is because the current standard secondary distribution voltage is 11kV; also it will be recalled that all of the surrounding area is supplied by an 11 kV secondary network. Although this will not automatically rule out alternative 4 it will give pause for

thought because it will conflict with the commitment to provide load transfer potential when possible. This commitment rests in turn on the more general commitment to maximize system flexibility[2].

New perspectives lead to refinement of earlier themes, alternative 3 is adapted to benefit from new insights and as alternative 3 ° it supersedes alternative 3

The replacing of the four old transformers with three similar ones (alternative 3) could be achieved by placing them on the site in such a way as to release part of the site for redevelopment provided that a new switchboard is installed with them.

A further alternative is suggested by this same problem framing. It takes the idea of installing a new sub-station to its natural conclusion given the design commitments referred to in the evaluation of alternative 4. The designer attends to the idea of uprating the network supplied by Branse (Int6.32)

Alternative 5

Install a new sub-station like that proposed in alternative 4 but with transformer secondary voltages of 11 kV rather than 6.6 kV. Uprate the secondary network supplied by Branse to 11 kV (Int6.23).

Evaluation of alternative 5 makes reference to alternative 4

The need for an autotransformer to ensure compliance with the security constraints for alternative 4 is unnecessary for alternative 5 since the secondary voltages of the distribution network supplied by Branse will be 11 kV, the same as that of the surrounding area. Hence interconnection will be simplified.

A final (third) problem framing is also offered. It might be seen as an approach that would be considered automatically (for it has been seen in the previous two examples of designing already). It is the view of the problem in terms of a need to transfer load to the surrounding network (and in this case, as a consequence, to close down Branse sub-station). However, the idea of seeing the problem this way for this project is not feasible until the idea of uprating the secondary network supplied by Branse is entertained. To this extent this final problem framing can be seen to follow on from the ideas brought to the fore by alternative 5.

Relevant properties of the surrounding network

Ignoring the differences in secondary distribution voltage, the load supplied by Branse could be met by the surrounding sub-stations.

Alternative 6

Uprate the secondary network currently supplied by Branse to 11 kV and supply it from spare capacity in the surrounding

area. Close down Branse sub-station (Int6.22, Int6.26).

Evaluation of alternative 6 shows how an alternative may subsume or eliminate another which may have originally given rise to it

Alternative 6 has the advantages of alternative 5 with some extra ones. If Branse is closed down the whole site could be sold off and the cost of installing a new sub-station will be avoided.

Design commitments bear on the evaluation

Geographical and electrical isolation of the area supplied by Branse mitigates against this since there would be "large areas of common land to cross" (Int6.22). System losses will be high since the area will cease to be supplied from a sub-station at the load centre[2].

1 The security constraints are shown in full in the form of a systemic grammar network in section 5.2.5 of this chapter.

2 These design commitments, their relationship to one another, and to others, is shown in the form of a systemic grammar network in section 5.2.4 of this chapter.

4.1.2 Linking the designing to the design recommendation

In the draft planning proposal associated with this work six schemes for handling the situation at Branse over the next twenty five years are considered. They correspond to the alternatives outlined above, sometimes proposed in combination to cover twenty five years ahead. For example, alternatives 1 and 2 are each combined with a proposal to carry out alternative 4 in ten years time. On the other hand alternative 4 is a scheme which could be carried out immediately as a solution which would "last" for twenty years (at least).

In the planning report it is argued that alternative 6 rules out alternative 5 essentially for the reasons given above (in 4.1.1). Alternative 2 is dismissed on the basis of a cost comparison with alternative 1. The extra cost which would be incurred to reconfigure the transformer connections at 66 kV is not deemed to compensate for the removal of the poorly performing gas filled cables. This leaves four possibilities. Alternative 6 is ruled out on the basis of technical unacceptability as argued above (in 4.1.1). The cost of losses over the twenty five year period would amount to 30% more than the cost of two of the other alternatives (3° and 4). Nevertheless alternative 6 is presented fully costed for comparison with the other three remaining alternatives.

Alternative 1 also appears fully costed for comparison (it also would cost about 30% more over twenty years that alternatives 3° and 4). This leaves

two realistic candidates which are presented as alternative 3° (followed in ten years by 4) in a comparison with alternative 4 (carried out at the outset of the twenty five year period). Alternative 3° is calculated to be about 10% cheaper over twenty five years than alternative 4 but involves higher initial capital outlay. Alternative 3° is recommended with stated reservations over the uncertainty associated with the performance of the second hand transformers and the gas filled cable which would be left in service under this proposal - "the failure in advance of the year 2002/3 of either the second hand replacement 66/6.6 kV transformers or the remaining gas filled cable ... would mitigate in favour of the adoption of" alternative 4 "conversely the continued satisfactory performance of all of these items beyond the year 2002/3 would favour the adoption of" alternative 3° (PR).

4.1.3 Designer's appreciation of the outcome

Due to changes in operating conditions (privatisation of the electricity supply industry and the consequential interim license conditions regulating the Regional Electricity Companies which replaced the nationalised Area Supply Boards) the proposal - alternative 3° was not carried out. Briefly stated, the reason for this was that under the licence conditions applying at the time of the work, benefit from resale of part of the site of Branse sub-station would not accrue to the company and therefore would not offset the cost of installing a new switch board. The design solution implemented therefore corresponded to design alternative 3, i.e. simply placing three second hand transformers on the site, connecting them in the cheapest manner to the existing switchboard, and therefore leaving the site fully occupied by the sub-station. The designer's appreciation given here refers to the work carried out (alternative 3) rather than to the recommended design proposal (alternative 3°).

The main problem was with the obsolete transformers; the poorly performing gas filled cables constituted an additional weakness. The work authorised to be carried out, namely the replacement of the obsolete transformers with second hand ones, satisfies the main cause for concern. The solution will last until there are further developments in the area or until other incidents necessitate major system redesign in the area. The matter of the aging gas filled cables remains. There is no flexibility (potential for load expansion, reuse of part of the site) in this solution. Replacement of the transformers was the cheapest solution; it does not lend itself to further developments or offer any longer term benefits (Int8.14). This appreciation is based on the repertory grid check exercise. The data used for the check gives a fuller description of the designer's view encompassing more of the alternatives considered as well as the outcome. It is presented in figures 6.6 and 6.7 in chapter 6.

4.2 Discussion

4.2.1 Evaluation increases situation understanding

The evaluation of design alternatives brings different aspects of the

design situation into focus. Evaluation of an alternative leads to a greater appreciation of the situation overall and may also result in a shift of emphasis. In this example we see that the idea of releasing some of the sub-station site for resale, an idea which comes about through the evolution of alternative 4, stimulates awareness of this issue in a way which then informs development and evaluation of (other) design alternatives. Alternative 3 is refined in the light of this appreciation. Thus, it can be seen that a problem framing does not only result in a set of solution alternatives which satisfy that problem definition but also contributes to understanding the situation generally. Insights gained from seeing the problem in different ways and evaluating potential solutions are not confined in application to the views through which they are initially generated.

4.2.2 Design commitments can selectively constrain the task

In the Branse Transformers example it is clear that the designer feels uncomfortable with the idea of installing new plant which operates at a secondary voltage of 6.6 kV. This unease is caused by the fact that this solution conflicts fairly comprehensively with all the qualitative design commitments (see 5.2.4 below). Firstly by restricting load transfers at the secondary distribution level, it goes against the commitment to maximize flexibility. Secondly, it perpetuates the use of intermediate transformation which in turn prevents reduction in the amount of equipment installed. Finally, it results in retention of the higher secondary system losses associated with 6.6 kV distribution networks so it makes no contribution towards reduction in system losses. To deliberately propose a design which will perpetuate the island of 6.6 kV network which mismatches with its surroundings which operates at 11 kV is seen as counter-intuitive on all of these grounds. Those who read the design proposal will share these prejudices. (This fact influences the presentation of the design proposal as shown below in section 4.2.3.) By contrast, the idea of uprating the site's secondary voltage is seen by the designer as a major (positive) theme directing some of his designing. He says, "there really are about three ways of coming at this (project) ... the uprating is a main option, and you've got variations on that main option ... that can be one approach to dealing with the problem" (Int6.32). The qualities the designer seeks in the design alternatives is affected by this idea. The designer makes additional demands of his designs as a consequence, so both what he generates by way of alternatives and how he evaluates them is affected. Several of the alternatives which he explores can be seen to be strongly influenced (constrained) by the attempt to include network uprating into what is proposed.

4.2.3 Designs are justified in relation to alternatives

The role of design alternatives in supporting the design alternative that is recommended is clearly shown for this example in section 4.1.2 and for the case of the Breedpace Lane Reinforcement project in section 3.1.2. In the case of the Branse Transformers the fact that designs are justified in relation to alternatives is very clearly demonstrated in the roles of alternatives 6 and 1 in the design proposal. Alternative 6 signals the

natural preference for uprating the secondary network currently supplied by Branse. (It is "natural" for the reasons given above in section 4.2.2.) Its presence in a full comparison with alternatives 1, 3° and 4 acknowledges the shared commitments motivating interest in uprating. Its appearance allays disquiet which might otherwise arise over the design proposed by showing the technical and financial infeasibility (in this case) of uprating. Thus the "omission" of uprating from the practical candidates for recommendation (alternatives 3° and 4) is justified by showing them as more favourable when compared with alternative 6.

Alternative 1 also plays a similar role in justifying alternative 3° and 4 by showing why the effort they entail is justified over the simpler option (simpler in the sense of causing the least disturbance or alteration to the existing network) of replacing the four obsolete transformers with four others which alternative 1 proposes.

5 Representing the Design Situation

The systemic grammar network presented and illustrated here captures the representable aspects of the design situation relevant for competence modelling. Firstly, an overview of the dimensions of the design situation which are expressed in the systemic grammar network is given (section 5.1). This is followed by fuller presentation of the network with brief descriptions of its main aspects (section 5.2). Finally, using examples from specific design projects, paradigms of the network are shown to illustrate its generative and descriptive powers (section 5.3).

5.1 Overview of the SGN Representation

The SGN constructed to represent the design situation consists of five aspects. These are shown in figure 7.1. Each aspect shown in detail and discussed in turn in section 5.2 below. Before this, however, an overview of the representation is given. Briefly, the top level dimensions, the five aspects, are as follows. Firstly there is a representation of the initial motivation for the design activity, labelled DesReqs. Secondly, there is a representation of the existing electrical distribution network, labelled NetCntx, from which the specific network context relevant for each design project can be generated. This part of the network can express all the aspects of the "environment" that the design activity must take into account, capturing the idea that design decisions will both affect, and be affected by, the existing electrical network.

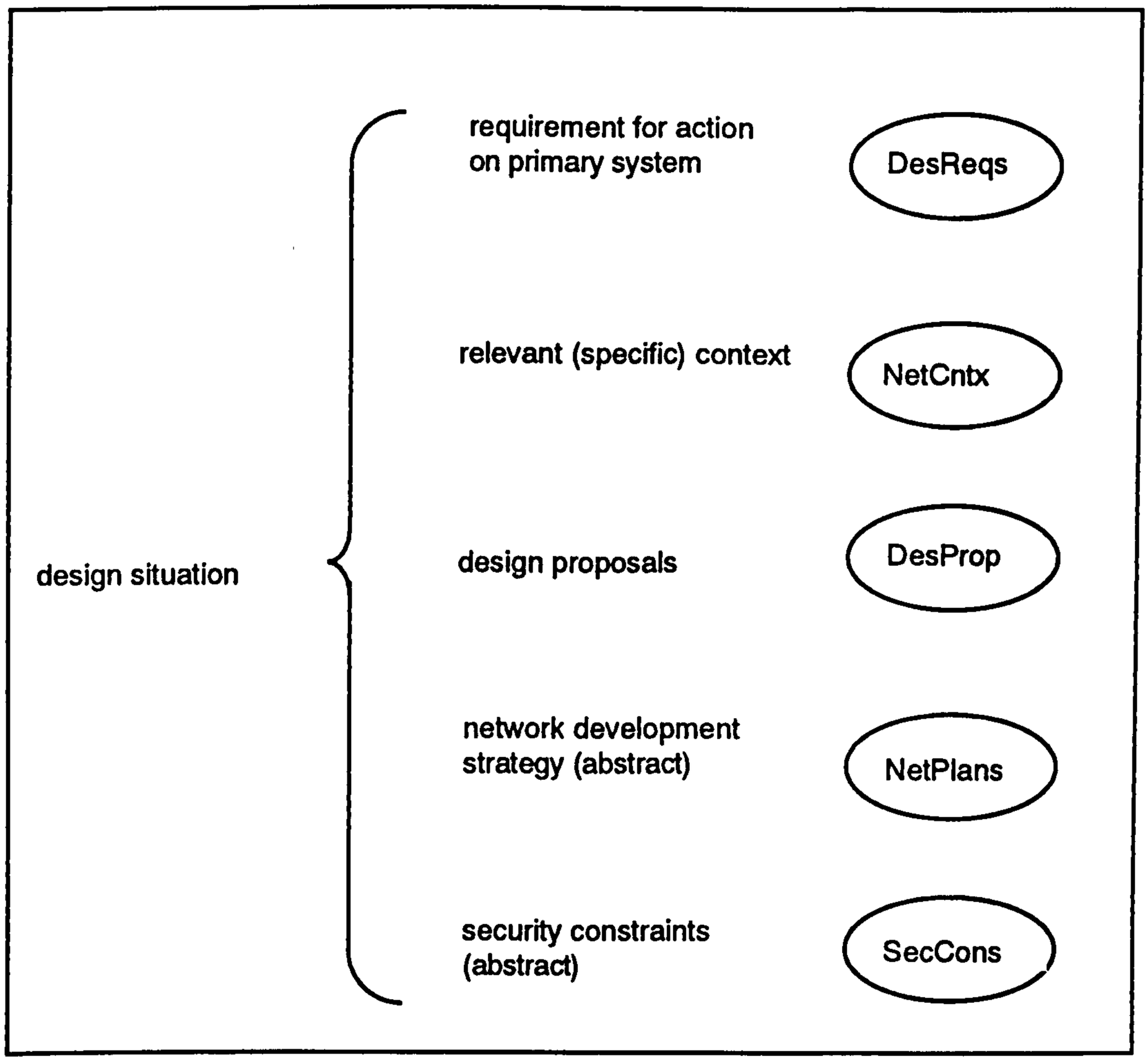


Figure 7.1 Overview of Design Situation.

Thirdly part of the SGN expresses the design recommendation which results from the design activity and the design alternatives that evolve during designing. This aspect of the design situation is labelled DesProp. The fourth and fifth components represent qualitative and quantitative design commitments. The qualitative one, labelled NetPlans, captures the design commitments, (abstract plans for how the network should be designed) which both motivate the designer to design in the way he does and which he, and the professional group to which he is accountable, use to judge the qualities of a design alternative and to compare the merits of design alternatives. The component labelled SecCons represents the quantitative, the "hard" constraints, the ones to which a design must conform to be acceptable. The representation of these is directly derived from the published standards for security to which each public electricity supply licence holder must conform (Electricity Association, 1978).

To summarize then, the representation of the design situation comprises a representation of the design proposal (DesProp) and the main constraining and enabling influences on its generation namely the initial circumstances motivating the designing (DesReqs), the context in which

the designing takes place and with which any recommendation must favourably interact (NetCntx), and the design commitments (NetPlans and SecCons).

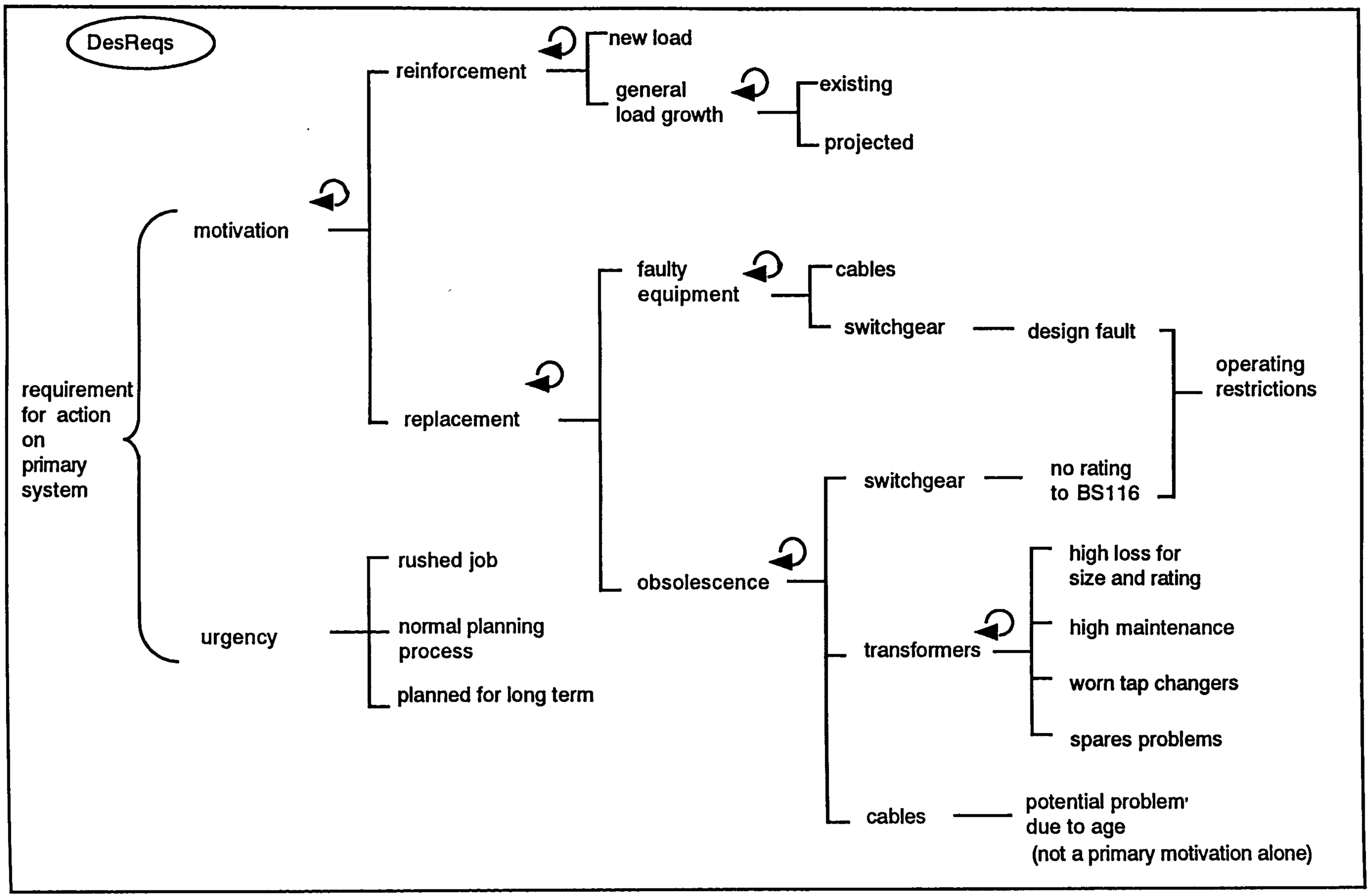
5.2 Systemic Grammar Network - Main Components

Each aspect of the design situation introduced above is now described in turn.

5.2.1 Initial motivation for action

The part of the network which captures the initial motivation for action (DesReqs in figure 7.1) has two aspects to it. These are the motivation for the designing and the urgency with which the designing (and more accurately the outcome of it) must be effected. DesReqs is shown in figure 7.2.

Figure 7.2 SGN showing Primary Motivation for Design (Requirements).



Motivation ought to be expressed in a form which does not pre-empt the form of the solution since this might unwittingly constrain the designer's approach unnecessarily. Use of the terms "reinforcement" and "replacement" might initially be thought to be ill-chosen from this point of view since they may appear to imply the form of the design solution. However these are the terms used by the designer and it is clear that use of them in classification does not constrain his consideration of alternative problem framings. Examination of the terms of increased delicacy associated with "reinforcement" and "replacement" (those shown to the right of them in figure 7.2) provides reassurance on this point particularly if they are compared with the terms "reinforce network" and "replace equipment" in the part of the network which deals with the design alternatives (labelled DesProp in figure 7.1, described in section 5.2.3 below and shown as figure 7.4).

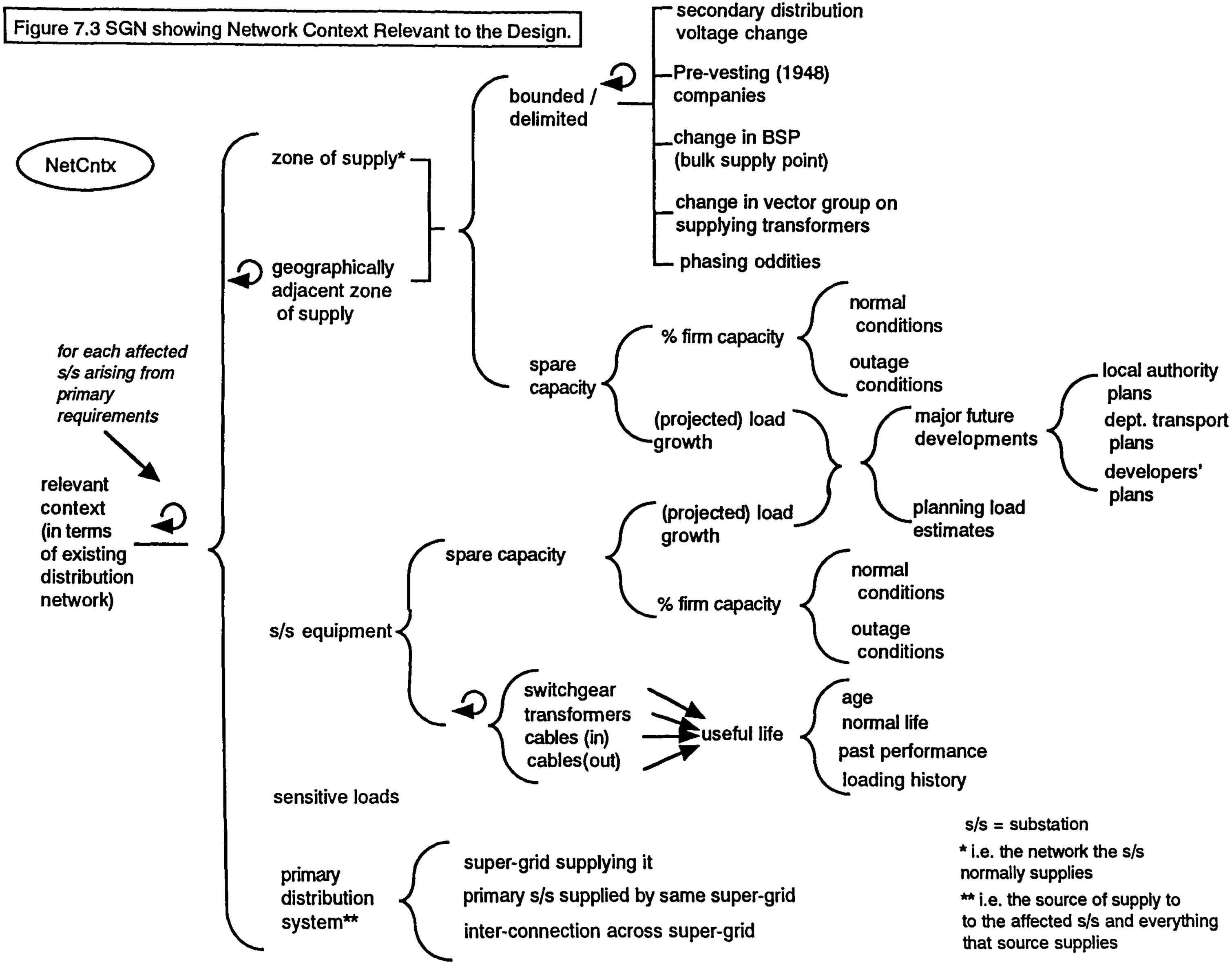
An example clarifies this point - one can satisfy the requirement to reinforce by replacing equipment by, for example, installing a larger capacity transformer in place of a smaller one. The distinction in DesReqs between reinforcement and replacement plays an important role for the designer because of associations with allocating costs of work proposed between capital and revenue budgets.

Most situations are not simply reinforcement or replacement. Example 1, the Swedenill Diversion, described above in section 2 was expressly chosen by the designer as the first project discussed during knowledge elicitation precisely because it was unusually straight-forward in this respect (Int1.2, Int2.9). The use of the SGN symbol for repeated selection from the terms "reinforcement" and "replacement" permits generation of expressions which capture the possible mixed combinations making up the primary motivation for designing.

5.2.2 Establishing the relevant (electrical network) context

The branch of engineering design which provided the material for this case study always concerns designing new or replacement components of the electrical distribution network. Knowledge of where the design must fit in and what parts of the network (and which parameters of it) will affect a design recommendation and be affected by it is a central concern. This aspect of the design situation is represented in the part of the network labelled NetCntx in figure 7.1 and is shown with its terms in figure 7.3.

Figure 7.3 SGN showing Network Context Relevant to the Design.



The sub-stations mentioned in DesReqs give the starting point for establishing the electrical network context relevant to the designing in each case. The context "fans out", as it were, through the electrical network, from these sub-stations. The two most important concepts here are the zone of supply, that is the network the sub-station normally supplies with electrical energy, and the source of supply to the affected sub-station(s) which is termed the primary distribution system. Electrical connectivity is not the only relevant perspective however since the physical geography of the network is also pertinent to the consideration of the full range of design possibilities. Thus the NetCntx part of the SGN makes reference to geographically adjacent zones of supply (as well as those which qualify for consideration through electrical properties).

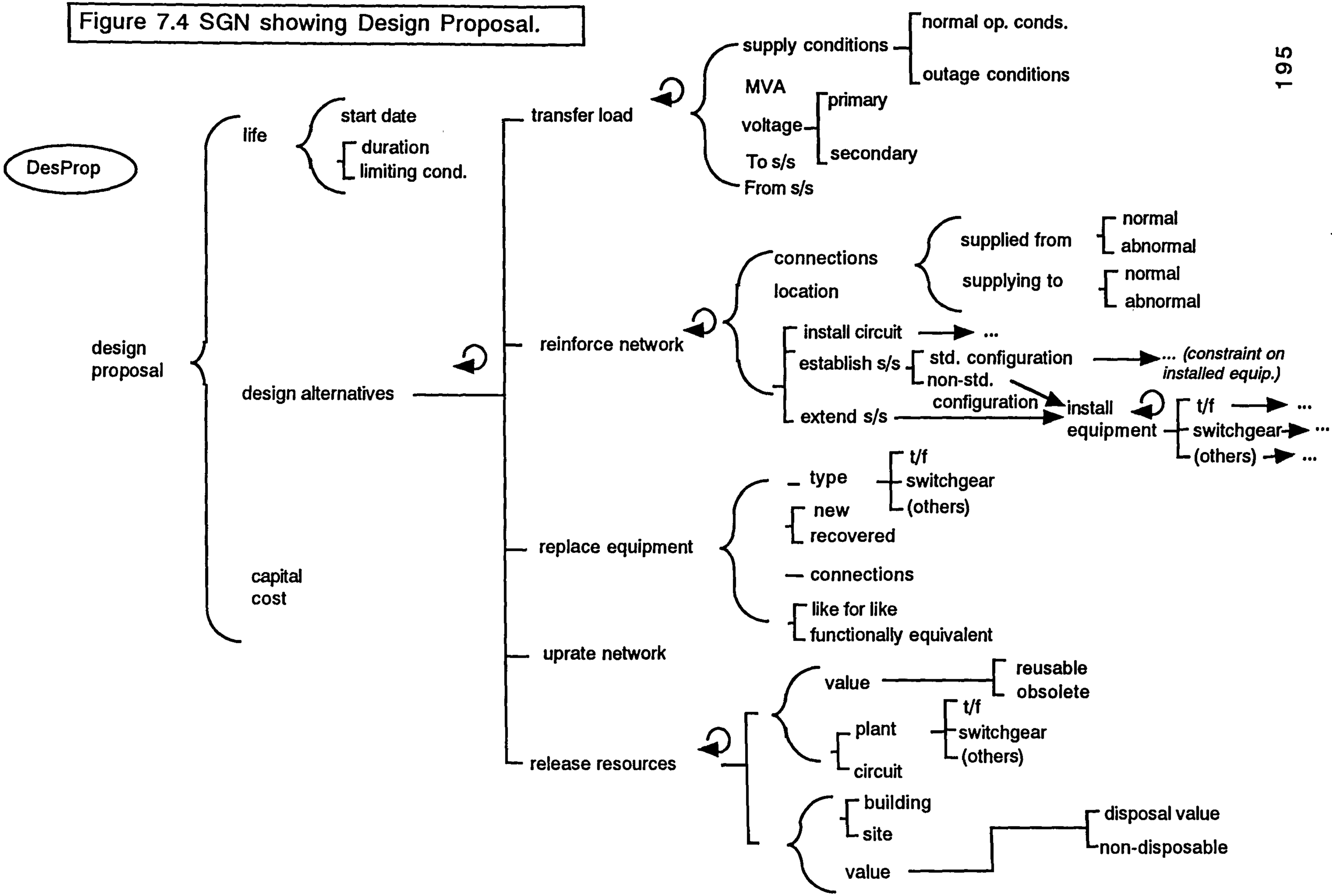
Having established which parts of the network form the relevant context for the designing it is also necessary to be able to express what it is about the identified network which is of interest, i.e. which properties the designer will wish to make use of in his designing. He may "use" them to advantage in his design proposals and/or he may have to accommodate them, that is, take them into account when designing. The main aspects of the network which are of interest are properties of the equipment installed at the sub-stations, properties of the cables which connect the equipment into circuits, and present and projected loading of equipment and circuits. The designer is usually interested in loading expressed in the form of spare capacity, and he will need to take account of electrical network properties under both normal and abnormal (outage) supply arrangements.

In densely built up urban areas the electrical network is usually interconnected on the secondary distribution side. This means that the boundaries of the zones of supply (see top right hand side of NetCntx, figure 7.3) are more complex to delineate than in a radially operated secondary network. (In smaller towns and rural areas the practice is to run secondary networks radially connected to their supplying transformers only. This simplifies many aspects of system management and routine operation, and incidentally makes the zone of supply simpler to identify.)

5.2.3 Design proposal

The part of the grammar which generates expressions of design alternatives is shown in figure 7.4.

Figure 7.4 SGN showing Design Proposal.



It can be used to express outline ideas for alternative design solutions of the sort presented in the accounts of the design projects given above in sections 2.1.1, 3.1.1 and 4.1.1. It is also powerful enough to express the salient aspects of a design solution in the detail to which it is expressed by the designer in a (final) design recommendation in the design proposal document. An example of this is given below as an illustrative SGN paradigm in section 5.3.2. A design proposal has three main components, a description of what is proposed, an associated cost, and a life expectancy i.e. when and for how long the proposal will be effective.

Each design alternative, one of which will be proposed by the designer for approval, will comprise a recommendation to transfer load, to reinforce the network, to replace part of the network, to remove plant or buildings from service, or to uprate the voltage of part of the network, or a combination of these. The recursion symbol shown in association with these choices in figure 7.4 indicates that they may be selected in combination with one another and thus illustrates how complex combinations of choices can be simply and clearly represented in an SGN. The network extending to the right of each choice shows how each one of the possibilities can be refined by further relevant categorisations increasing in delicacy. The network shows what the further choices are in each case and how they may be combined.

It is worth recalling here that, as stated in chapter 3 the analyst in constructing a SGN to present a view of qualitative data, is constructing a language to suit a purpose. The network show similarities and differences which make relevant categorical distinctions. Here, for example, a distinction is made between, on the one hand, reinforcing a network by extending a sub-station or establishing a new one and, on the other hand, by replacing equipment, for example by functionally equivalent (but different) plant. This distinction is one which makes sense to the designer. It can be related in a simple way to cost allocations but the costs distinction also has a more fundamental influence by affecting what the designer entertains as possible. The source of funding affects what the designer considers doing: he knows that what can be justified (in terms of expenditure) differs according to the source of funding.

The categorisation presented here generates plausible design alternatives which make sense to the designer. Its merit lies solely in this; other representations are not therefore ruled out by acceptance of the value of the ones chosen here.

5.2.4 Qualitative design commitments - network development strategy

A proposed design must deal satisfactorily with the initial "problem" or circumstances that motivate the design activity in the first place. It must fit in with the existing electrical network, not adversely affecting it, and making improvements to it when possible. The design must also meet legal or other regulatory constraints. The important formative constraints of this kind for the case study design environment are security constraints which are discussed in the next section. There are , however, other "soft"

constraining (and enabling) influences on the designing which are derived from the norms and practices, tacit or overt, of the design engineers, operational engineers and system managers who develop and operate the electricity distribution network.

These constraints provide an underlying agenda which guides the designer to design in certain ways, and to both identify and value certain qualities in a design. These are the qualitative design commitments. They both inform and direct the designing and play a key role as qualitative evaluation criteria for judging design alternatives and for comparing one with another. The designer may even make reference to these design commitments explicitly in some form when presenting the case for a design proposal. He uses them in justifying his design proposals. Examples of qualitative design commitments used in this way have already been shown in the accounts of designing for the three examples given above in sections 2.1.1, 3.1.1, and 4.1.1. The commitments referred to in those examples are represented along with the others which influence the designing in the part of the SGN shown as NetPlans in figure 7.1. NetPlans is shown in detail in figure 7.5.

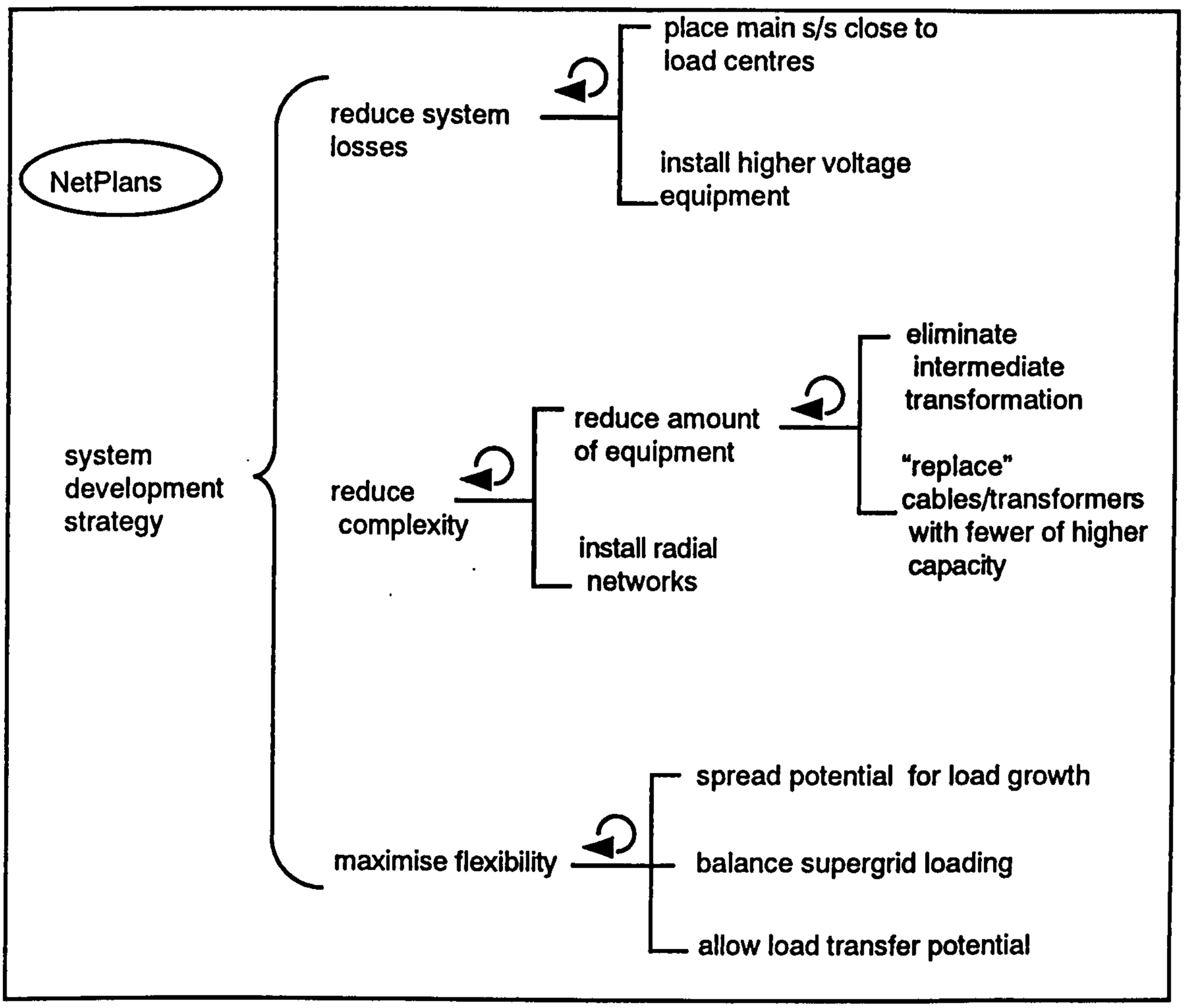


Figure 7.5 SGN showing Qualitative Design Commitments - Network Development Strategy

5.2.5 Quantitative design commitments - security constraints

The primary standard against which the quality of the distribution network is judged and to which its operators must conform is laid out in an engineering recommendation originally produced by the Electricity Council Chief Engineers' Conference and currently published by the Electricity Association as Engineering Recommendation P.2/5 (Electricity Association, 1978). The Electricity Act of 1989 (HMSO, 1990) states that "the licensee shall plan and develop the licensee distribution system in accordance with a standard not less than that set out in Engineering Recommendation P.2/5 ... in so far as applicable to it or such other standard of planning as the licensee may, following consultation ... adopt from time to time" (HMSO, op.cit., p.96). The part of the SGN labelled SecCons in figure 7.1 is shown in detail in figure 7.6.

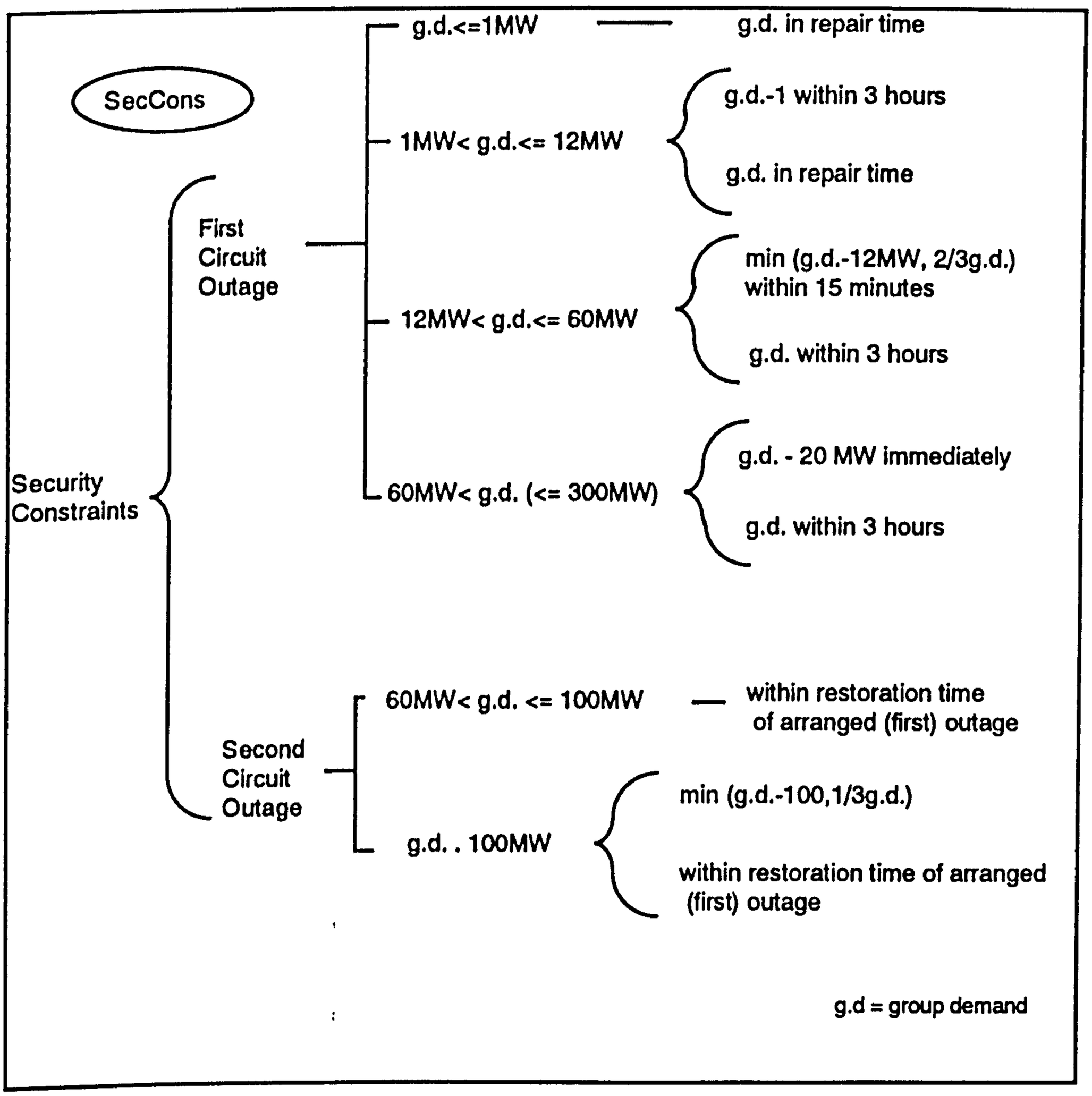


Figure 7.6 SGN showing Quantitative Design Commitments - Security Constraints.

SecCons is a representation of the security standards as published in P.2/5 and as such it is limited in value for understanding the influences on the designer in two ways. Firstly, these regulations, like any others, are open to interpretation (Int2.26, Int6.19-6.21). As the designer puts it "there are get out clauses" (Int2.26) which can be interpreted. One example of this has been given (the Swedenill Diversion example described in section 2.1.1 and discussed in section 2.2.2). In that example the time to travel between sub-station sites is a factor in determining whether or not electricity supplies can be restored within regulation times. Compliance with the security standards, or rather judgement as to whether compliance can be effected, is not solely an electrical property of a distribution network. There are legal responsibilities at work here but also the professional margins for interpretation inevitably associated with regulations. The representation given in SecCons is restricted to reference to electrical network constraints only (Int 4.13) for reasons which lie to some extent with the second limitation.

The second limitation with SecCons results from the following circumstances. Hypothetically, a licensee may design parts of the electrical distribution network to higher standards than those laid out in P.2/5 for operational reasons and by way of exercising the professional margin associated with legal requirements. However, any such standards, if published or acknowledged in any way, would render the licence holder open to liability, claims for compensation, and so on for any breach of a stated intention to operate to those standards. This is a situation the licensee would wish to avoid. Despite not being publicly acknowledged, any agreement among the design engineers, operational engineers and system managers to conform to a standard other than P. 2/5 would obviously affect the way networks were designed. That which is represented in SecCons (only) reflects the published regulations on security of supply, should there be any tacit agreement to work (design) to other standards, attempts to understand how a design is arrived at would only be understandable on the basis of knowing the standards being used.

There is an important general point to be noted here. There are inevitable limitations to what can be represented in a knowledge based system which arise not from the inherent limitations of any representation medium but from issues of confidentiality and from the legal implications of any explicit representation whether in a policy document or in a knowledge base.

Returning to the specific situation of this case study the security constraints, as they are expressed in SecCons, are unsatisfactory for this reason. However, a representation of them as defined in P.2/5 is given for completeness, the form they take is briefly as follows. According to P.2/5, the security constraints are expressed essentially in terms of how quickly load which is disconnected from supply due to fault, damage or any other abnormal circumstance must be resupplied. Examination of SecCons (figure 7.6) shows that longer disconnection periods are tolerated for smaller loads and that the situation for a single cause of abnormality is

distinguished from the situation when two events affecting the same load occur simultaneously; these are first and second circuit outages respectively. The quantitative security constraints imposed on designs always take precedence over the qualitative constraints represented in NetPlans (Int3.12).

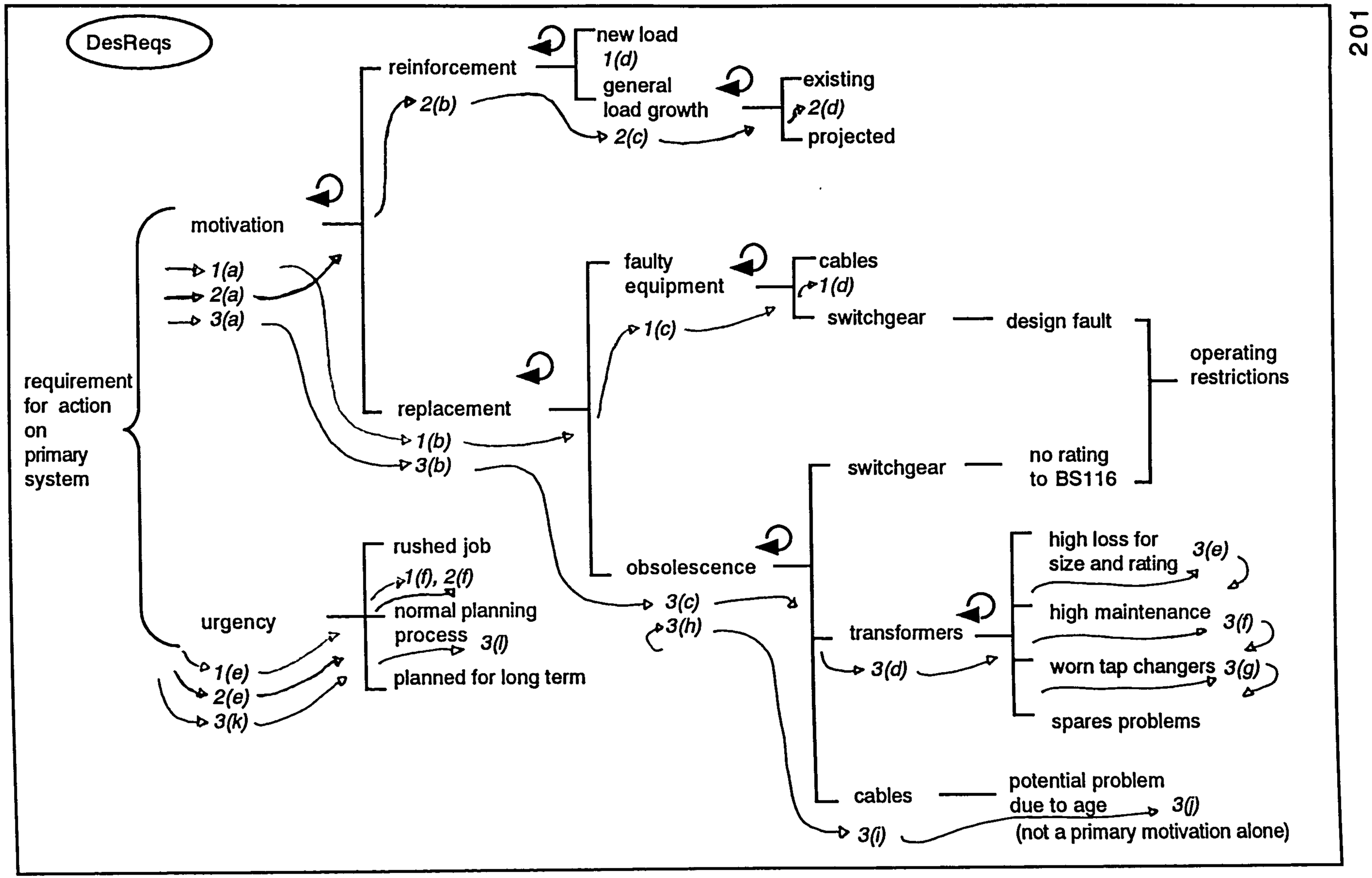
5.3 Illustrations of Paradigms

In this section illustrations of how instances of initial motivations for design and a design proposal constitute paths through the relevant parts of the SGN (based on DesReqs (figure 7.2) and DesProp (figure 7.4) respectively) are given.

5.3.1 Illustrations of initial motivations for design

Three examples are given here of how expressions of initial motivations relate to the systemic grammar given by the part of the SGN labelled DesReqs. The descriptions of initial circumstances given above for the three examples of design projects (sections 2.1.1, 3.1.1, and 4.1.1) can be seen as paths through the grammar represented by DesReqs. Below each description of initial circumstances is reproduced for each of the three examples with annotations to show how each one constitutes a path through the terms of the grammar. Figure 7.7 shows the paths superimposed on DesReqs. Paths are labelled with the example number (1, 2, or 3) and letter sequences. Thus, the description of example 1 can be understood using the terms in the path labelled 1(a) through to 1(f), example 2 by following 2(a) to 2(f) and example 3 by following 3(a) to 3(l).

Figure 7.7 SGN showing Primary Motivation for Design (Requirements) annotated with example paths.



Example 1 - the Swedenill Diversion

A case of faulty cables motivates (1(a)) the requirement for action. The requirement to deal with faulty equipment is regarded along with handling obsolete equipment as "replacement" for cost attribution purposes (1(b)) - source : Int3.1-3.2.

"A particular 66 kV cable route consisting of four cables supplying a 66/11 kV sub-station with a load of 40 MW has become a problem. The number of faults (1(c)) occurring on the cable (1(d)) route is abnormally high and increasing." (description of example 1 given in section 2.2.1 above) - source: Int1.2, Int1.4, and planning proposal.

A design was needed quickly since action to tackle the faulty cables needed to be carried out in a hurry (1(e) & 1(f)) - source : repertory grid exercise (the Swedenill Diversion corresponds with element labelled A in appendix 3).

Example 2 - the Breedpace Lane Reinforcement

A case of overloading on part of the electrical network motivates (2(a)) design activity to reinforce the network (the need to deal with load growth is a kind of reinforcement (2(b))) - source: Int3.1-3.2.

"An area of the network is becoming overloaded due to general load growth in the locality (2(c)). The overloading has been exacerbated by the decommissioning of a 132/6.6 kV sub-station two or three years previously ... which ... had previously fed some of the load in the area (2(d))." (description of example given in section 3.1.1 above) - source: Int2.13, Int2.15, Int2.30.

A design solution to handle the circumstance was needed quickly (2(e) & 2(f)) - source : repertory grid exercise (the Breedpace Lane Reinforcement corresponds with the element labelled B in appendix 3).

Example 3 - the Branse Transformers

A case of obsolete transformers motivates (3(a)) design activity initially.

"A sub-station, Branse, supplying a geographically and electrically isolated area provides energy via four 15 MVA 66/6.6 kV transformers. These transformers have reached the end of their natural life (3(c) & (d). They were installed sixty years previously and now are leaking oil to an unacceptable degree, the tap changers are obsolete and break down frequently (3 (g)), and revenue costs (3(e)) associated with the transformers is about seven times the cost of maintaining other equivalent plant (3(f)) ...

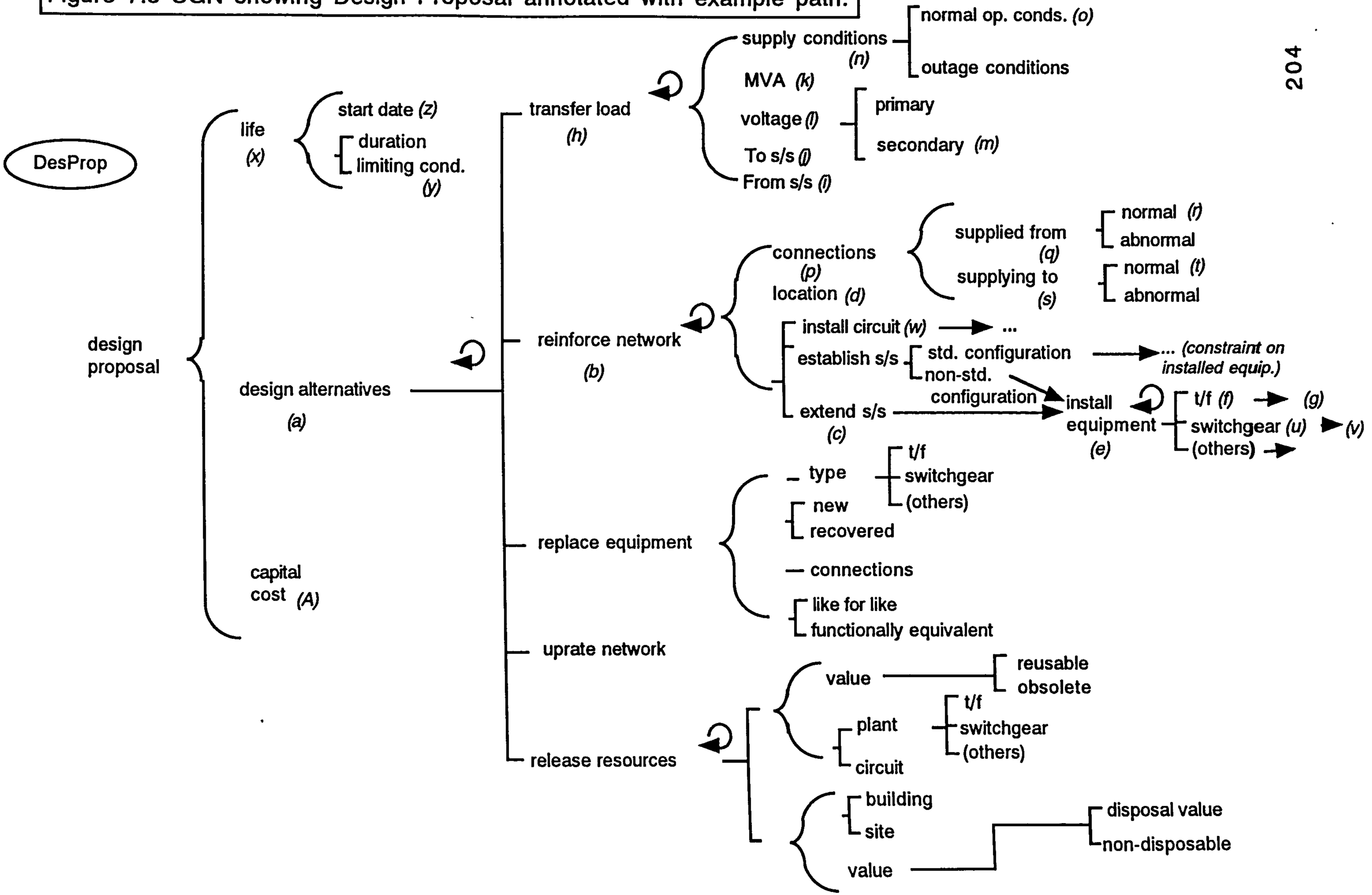
The old transformers are of very low impedance, as a result reactors have been installed at Branse to keep the fault level down to an acceptable level (3(f)) ... energy supply to the transformers is via four cables, two of which are oil filled cables in good condition, two of which are 40 year old gas filled cables (3 (h) & (i)) of an age where leakage is expected to be a problem. Some poor performance has already been noticed on the gas cables circuit ... after about 30 years of service gas filled cables start to leak, life expectancy is about 40 years (3(j)) ... no load growth is expected in the next few years (3(b) (only))." (description of example given in section 4.1.1 above) - source : Int6.8-6.12.

The problem has developed gradually, it can be handled in the normal timescales of design projects (3(k) & 3(l)) - source : repertory grid check see figure 6.6 in chapter 6).

5.3.2 Illustration of design proposal

In this section another part of the design situation SGN is used to demonstrate and emphasize the generative power of the SGN. The part selected is that which represents design proposals (labelled DesProp in figure 7.1 and shown in full in figure 7.4). This illustration gives an indication of how a rich and complex description can be generated from the compactly represented set of distinctions which comprises the grammar. To introduce some variety and to extend the range of examples a fourth project is introduced here. This fourth example concerns a proposal to extend an existing sub-station as a fairly short term measure to relieve overloading on part of the electrical distribution network. The summary of the design proposal for this project is reproduced below *exactly* as it appears in the design proposal document except that the sub-station names have been changed. It forms the left hand column. DesProp is shown in figure 7.8 annotated with the path through the terms in the network which generates the design proposal description. A commentary to the path is given in the right hand column below, the letters in brackets correspond with those marked in figure 7.8.

Figure 7.8 SGN showing Design Proposal annotated with example path.



DESIGN RECOMMENDATION	SGN PARADIGM
"This paper sets out proposals to install the fourth 15 MVA 33/11 kV transformer at SubOne sub-station	The design alternative proposed (a) that the network is to be reinforced (b) by extending (c) SubOne sub-station (location - (d)) by installing (e) a transformer (f) as specified (voltage, capacity, etc - (g)) connections information comes later in the text to complete this bracket .
to enable SubTwo sub-station to be relieved of 12.3 MVA of load by means of secondary distribution load transfers from SubTwo to SubThree,	Load is to be transferred (further selection from options to right of (a)) - (h) from SubTwo (i) to SubThree (j). The amount to be transferred is 12.3 MVA (k) at secondary voltage (l) & (m). It can be inferred from the sentence which follows that this is a transfer for normal operations (n) & (o).
and from SubThree to SubOne.	Load is also to be transferred (h) (repeated entry to bracket to right of transfer load) from SubThree (i) to SubOne (j). The amount to be transferred is 12.3 MVA (k) at secondary voltage (l) & (m), and again it is a transfer for normal operations (n) & (o) .
The proposals given are short term measures until the new CitySub sub-station is commissioned in the City Centre Square area programmed for 1991/92.	This with a later statement relates to limiting conditions of project life (x) & (y). (See asterisk later in this column.)
The transformer would be connected onto an existing idle 33 kV oil filled cable installed between SubFour and SubOne.	Connection to the new transformer (completing the bracket to right of reinforce network entered above for the sub-station extension) - this statement gives connection information (p), supply is from SubFour (q) to SubOne (s) for normal operation (r) & (t).
A one panel extension would be required to the 33 kV switchboard at SubFour,	More network reinforcement in the form of sub-station extension is to be effected (b) (repeated entry to bracket to right of reinforce network) . Sub-station extension (c) at Subfour (d) by installing (e) a switch panel (u) as specified - 33 kV, etc. (v) is to take place. Connections for the switch panel can be inferred (p), infeed will be from SubFour (q) & (r) and will be

	supplying SubOne.
and a seven panel 11 kV extension required at SubOne.	There is further network reinforcement in the form of sub-station extension (b) (repeated entry to bracket to right of reinforce network) . Sub-station extension (c) at SubOne (d) is to take place by installing (e) a switch panel (u) as specified - 11 kV, etc. (v) . Connections for the switch panel can be inferred (p), infeed from SubThree (q) & (r) (from original load transfer remarks, as is the fact that it will be supplying SubOne (s) & (t).
The transformer and 33 kV switchgear would be taken from spare stock. The 11 kV switchgear extension would be new equipment.	These two sentences complete the brackets to the right of install equipment for the three sub-station extensions (terms of increased delicacy to the right of transformer (t/f) or switchgear, as appropriate assuming recovered or new is recorded as an attribute).
Five new 11 kV interconnectors would be required between SubOne and SubThree to facilitate the 11 kV load transfers between these sub-stations.	More entries are made to the bracket to the right of reinforce network and connections come from load transfer statements. Locations are SubOne and SubThree respectively and paths pass through install circuit (w).
The SubTwo to SubThree load transfer would be effected using existing interconnection.	
Project completion is programmed for October 1987. The total estimated cost is £383,982."	This, with information earlier (y) * referring to CitySub, gives start date (z) and limiting condition (y) and with statement of cost (A) completes leftmost bracket of DesProp.

6 Review

The case study material has been analyzed and presented in this chapter in two complementary ways. The descriptions of the design process and its outcome for three specific examples of design projects of increasing complexity have been presented. Following the descriptions of each example the aspects of designing which were most salient or best

exemplified were discussed. In this way different perspectives have been highlighted in an attempt to explore different viewpoints from which to understand the designing described in all three examples.

The systemic grammar network (SGN) representation of the design situation is a more comprehensive representation of the knowledge elicited from the designer. The SGN is intended to define a grammar which captures the representable aspects of the design situation relevant for competent designing. An overview of the systemic grammar network was followed by more detailed presentation of each of its main components. Finally, the representational and generative powers of the SGN have been demonstrated using parts of the three examples of constructed descriptions presented earlier. The objective of section 5.3 has been to show how parts of specific design situations can be seen as paths through the systemic grammar network.

In the next chapter a sub-set of the material presented here is used to illustrate how some of the components of a knowledge based system to support this design domain can be implemented on a suitably powerful knowledge engineering environment.

CHAPTER 8

Mapping Components of the Architecture to a Software Environment

"Ça ira."
Benjamin Franklin

The purpose of this chapter is to indicate how some of the results of the analysis and interpretation of data, gathered as described in chapter 6 and analyzed and interpreted as described in chapter 7, can be mapped onto some of the components of a knowledge based system to support design. One objective is to demonstrate that some of the essential architectural components can be represented and the interaction between them can be supported on the platform provided by a general purpose knowledge engineering environment. The main objective, however, is to show that the inclusion of the components described within a knowledge based system gives it certain of the abilities it needs to be able to support design. The implemented components described here are not sufficient to support all of the design activity but they contribute necessary features to a knowledge based system for supporting the particular aspects of design on which this thesis has concentrated. An important aspect of what is presented here is to show how high level architectural components of a design support system can be readily implemented, in an orderly manner, on a suitably rich and flexible, but nevertheless general purpose, knowledge based system development platform.

Two decisions have affected what has been implemented. The first was to focus on components salient to the support of design activity as it has been characterized in part 1 of this thesis. The second was not to re-implement the components of the Competent Systems architecture described and presented implemented in Lisp by Keravnou (Keravnou, 1986).

The components presented here and the relationships between them can be described and inspected from both a static and a dynamic point of view. To support this inspection a *developer's* interface provides a means for invoking dynamic behaviour for the purposes of observing and inspecting the representation both statically and dynamically. The interface is *not* intended for use by a designer to interact with any design support system that the knowledge based system components implemented here would underpin. The form of the interface, i.e. what can be seen, demonstrates the *structure* of designer/user - system interaction and the interaction among the architectural components which results in the presentation of relevant material to the designer in a timely fashion as a consequence of the underlying architectural components. The way in which material is

presented however, i.e. the interface the designer would see, has not received attention. The emphasis is on the *underlying* architectural components capable of supporting the designer which allow him to design in a natural way. Once these are in place, the cosmetic appearance of the interface can be given attention separately, informed by guidelines and good practice from the interface design literature, in consultation with the designer/users. These non-structural aspects of the designer-system interaction have not been the subject of study in the work presented here.

Section 1 below briefly discusses the use of a knowledge engineering environment based on the object-oriented paradigm. Attention is focused on the suitability of such a platform for evolutionary development and the support an object-oriented approach to representation offers when implementing a knowledge based system based on a model with complex structure. In section 2 the key architectural components of a knowledge based system to support some aspects of designing are illustrated by showing how design alternatives can be linked to design commitments (section 2.1) and how design activity can be used to relevantly focus the design context (section 2.2). Representations for recording what design decisions have been considered which are suitable for supporting review and comparison of design alternatives are illustrated with comment on the bearing this representation has on supporting justifiable design recommendations (section 2.3). The chapter and the case study of part 2 of this thesis concludes with a discussion of the role of a knowledge based system to support primary distribution network planning within planning support as a whole (section 3.1). The value that knowledge elicitation and knowledge representation such as that reported in chapters 6 and 7 has per se - as a means of preserving and passing on design expertise independently of any value these activities have as prerequisites for knowledge based design support system implementation - is discussed in section 3.2.

1 Using a Knowledge Engineering Environment Based on the Object-Oriented Paradigm

The description given in this section focusses on the characteristics which render the object-oriented paradigm pre-eminently suitable for knowledge based system development and in particular on its suitability as a platform for the architectural components of a knowledge based system to support design. It is not intended to provide a comprehensive, general review of the object-oriented paradigm. The specific implementation platform used for this case study is KAPPA-PC. An overview of the facilities it provides to an application developer and its hardware and software support requirements are briefly outlined in appendix 4.

When an object-oriented view is taken towards the organization and development of software the software is seen as a collection of objects, each object consisting of both data structure (here effected as slots in an object)

and behaviour (here effected as methods associated with objects). The characteristic aspects of the object-oriented approach to representation which are most relevant here are the notions of classification, polymorphism, and inheritance, each of these can be briefly described as follows.

Objects with the same attributes (slots) and behaviour (methods) are grouped into a class. An object class is therefore an abstraction - a description of attributes and behaviours that usefully characterize a group of objects for a particular application. Any particular object will be an instance of the class to which it belongs.

Objects perform operations when they receive instructions (messages) - usually from other objects - it is the methods associated with a object which carry out the operations. Object-oriented operators can be polymorphic since as each object may have different methods for achieving the operations it is capable of supporting it can accommodate operating on different classes of objects.

Classes of objects can be defined in a hierarchical relationship to one another. A class of objects defined in general terms can be refined into successively more specialized classes (subclasses) each of which inherits the properties (attributes and behaviours) of its superclass to add to its special properties. Inheritance mechanisms remove much repetition from object specifications.

These aspects of the object-oriented view of software organization lead to valuable qualities in systems developed on an object-oriented basis. Once again, only those qualities most relevant for the work presented here are reviewed. Potential to make use of abstraction is well supported in object-oriented systems. The object-oriented approach encourages the placing of emphasis on what objects are (attributes) and what they do (behaviour) rather than on how they are implemented. Commitment to low level details is deferred. It has been argued that "proper use of abstraction allows the same model to be used for analysis, high-level design, program structure, and documentation" (Rumbaugh, 1991). The ability to focus attention away from low level detail is enhanced further by the support for information hiding which an object-oriented approach offers. The external aspects of an object - the operations it can perform - are defined separately from details of how the object performs its operations. Details of how a method to perform an operation is implemented are encapsulated in the object with which it is associated. Applications (or other objects) can make use of an operation offered by an object without being affected by how the object performs the operation. This not only means that low level detail is hidden but also, very importantly, that methods associated with objects can be changed without affecting the applications that use them.

In the object-oriented approach data and behaviour are combined in object definitions. The emphasis is on the structure of the objects rather

than on procedural structure. Parallels have been drawn with the ideas behind the use of data modelling as the basis of database design - the argument there being that the data used in an application is less subject to change than the way it is used. In an object-oriented approach to application development, because functionality is built up from object structures, the resulting application's structure is more flexible (adaptable to change) than it would be in an approach which emphasised procedural aspects e.g. functional decomposition as the organizational basis of the implementation.

1.1 Implementation Supported by a Computationally Rich Software Environment

The requirements for implementing a design support system demand a representation platform which is sufficiently flexible and rich enough to support implementation of a complex model in which the components interact with one another in a variety of complex behaviours. The object-oriented paradigm enables this sort of complexity to be achieved without either loss of control as system functionality is increased or chaos arising over flow of control. To reinforce this point the object-oriented approach can be contrasted with a top down approach to system design and implementation.

The top down approach promotes close adherence to some initially completely identified specification. In a good top-down design the elements of the decomposition (in this case the architectural components of a knowledge based system) are narrowly focused to meet the (pre) specified requirements (Meyer, 1988, pp.46-47). The object-oriented paradigm, by contrast, although no less organized than a top down decomposition, supports representation of a much more complex structure and most importantly allows the structure to be evolved as requirements emerge or become clear. This important property of the object-oriented approach arises as a result of the nature of object-oriented software architectures and the notion of services (operations) being provided by objects for use, combination, extension, etc., by others, as outlined above. Christopher Alexander in "A City is not a Tree" (Alexander, 1988) puts it thus

> "it must be emphasized , lest the orderly mind shrink in horror from anything that is not clearly articulated and categorized in tree form, that the idea of overlap, ambiguity, multiplicity, of aspects .. are not less orderly than the rigid tree, but more so. They represent a thicker, tougher, more subtle and more complex view of structure" (pp.75-76).

Alexander's description applies to the structure of the sort of model that needs to be implemented and at the same time characterizes the representation platform that an object-oriented software development environment should be able to provide.

1.2 Support for Incremental Implementation

In designing knowledge based systems for design support it is not only a requirement to be able to represent a complex model but also to be able to build up that complexity in an incremental manner. The case for evolutionary development of any software system where the requirements cannot be completely and unambiguously pre-specified has been argued convincingly elsewhere (Gilb, 1988; Crinnon, 1991). Using a platform which supports an object-oriented approach gives a number of well-established advantages vis-a-vis the evolutionary approach, a key benefit of object-oriented software construction is that it makes the software easier to modify. By building a system from objects which are implemented in a manner which supports encapsulation (information hiding), the features of objects, i.e. their attributes and behaviours, can be added in a progressive, incremental fashion (Meyer, op.cit. p.326). Modification of the objects has a limited impact on the system as a whole,

> "because of the highly decentralized nature of object-oriented architecture, repair of omissions or mistakes and enhancement is often achieved without much impact on the other classes (of objects already developed" (Meyer, op.cit., p.326).

In addition to this characteristic of the object-oriented approach which supports evolutionary development, there is the associated phenomenon that classes of objects naturally lend themselves to reuse and hence become attractive as the basis of a succession of prototypes. Decisions taken early on tend to be less critical to the determination of system structure in an object-oriented approach than they are in a situation in which functionality is developed top down, this is because in an object-oriented approach rigid sequencing constraints are not (inherently) imposed. In an object-oriented implementation the facilities offered by an object are there to be made use of by other objects in whatever order, for whatever purpose is most appropriate.

1.3 Qualities of the Resulting Software

The architectural structure of a knowledge based system to support design is determined to a significant extent by the requirements for explanation and justification of its suggestions and reasoning. It has already been established that this demands that the knowledge representation be explicit in certain ways (hence the system's static qualities) and that the reasoning be open to inspection (hence the system's means of representing and recording the dynamic behaviour). An object-oriented implementation platform supports achievement of these requirements mainly through its facilities to clearly retain, in the implemented software, the abstractions used in the model it represents. This quality, of openness to inspection, is exemplified in the mapping of parts of the model of design competence to an object-oriented implementation platform illustrated below in section 2.

2 Illustrations of Architectural Components with Reference to the Design Activity They Support

The mapping of some of the components specified in chapter 7 to a software environment are described in this section. The source code relating to the parts of the implementation described is given in appendix 5. The *developer's* interface which is mentioned in the introduction to this chapter is purely intended to provide a means for invoking the dynamic behaviour of the components described below and for inspecting their representation both statically and dynamically. A demonstrator's script for providing a demonstration of the software components is provided as appendix 6. Figure 8.1 shows the object classes implemented and their relationships to one another.

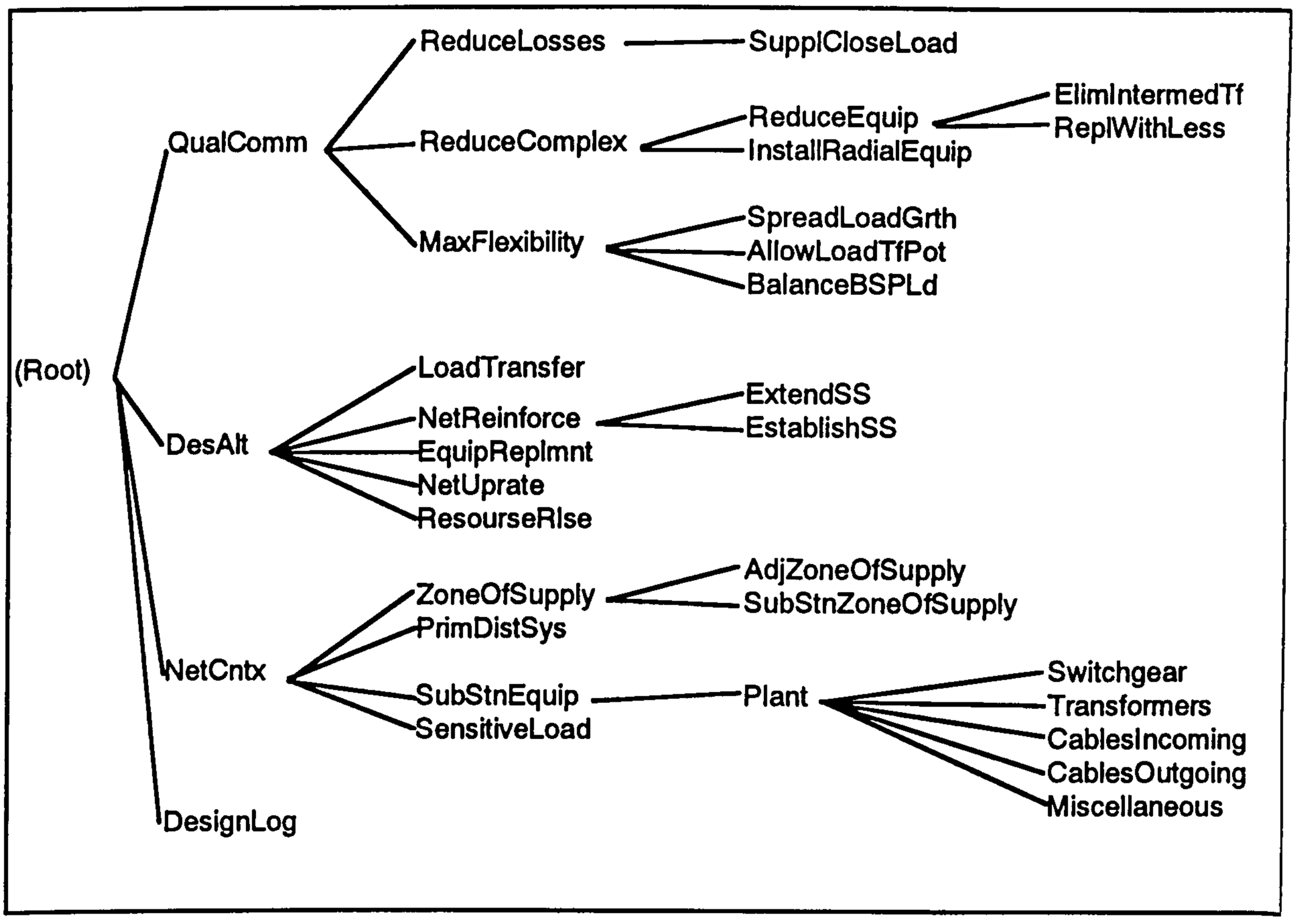


Figure 8.1 Implemented Object Hierarchy.

2.1 Representing Design Alternatives and Design Commitments and Making Links Between Them

The mapping of the systemic grammar network shown representing the design proposal as figure 7.4 and the mapping of the representation of qualitative design commitments shown in figure 7.5 (both figures are in chapter 7) and the way relevant links are made between them are described below.

2.1.1 Design alternatives as an object hierarchy

The design alternatives set out in figure 7.4 of chapter 7 can be represented as a hierarchy of objects in which each object represents a class of design alternatives. Slots, usually representing attributes of each object, and methods, each of which represents a behavioral aspect of the object, constitute each object. Both slots and methods are routinely inherited by sub-classes in the object hierarchy unless this is not appropriate. Attributes (slots) and procedures (methods) are defined at the most generally applicable level in the class hierarchy for the reasons outlined in section 1 above. An outline of the static view of the representation of design alternatives as a hierarchy of object classes, the "DesAlt" hierarchy, is shown in figure 8.2, a close structural relationship to the terms in the SGN of figure 7.4 can be observed.

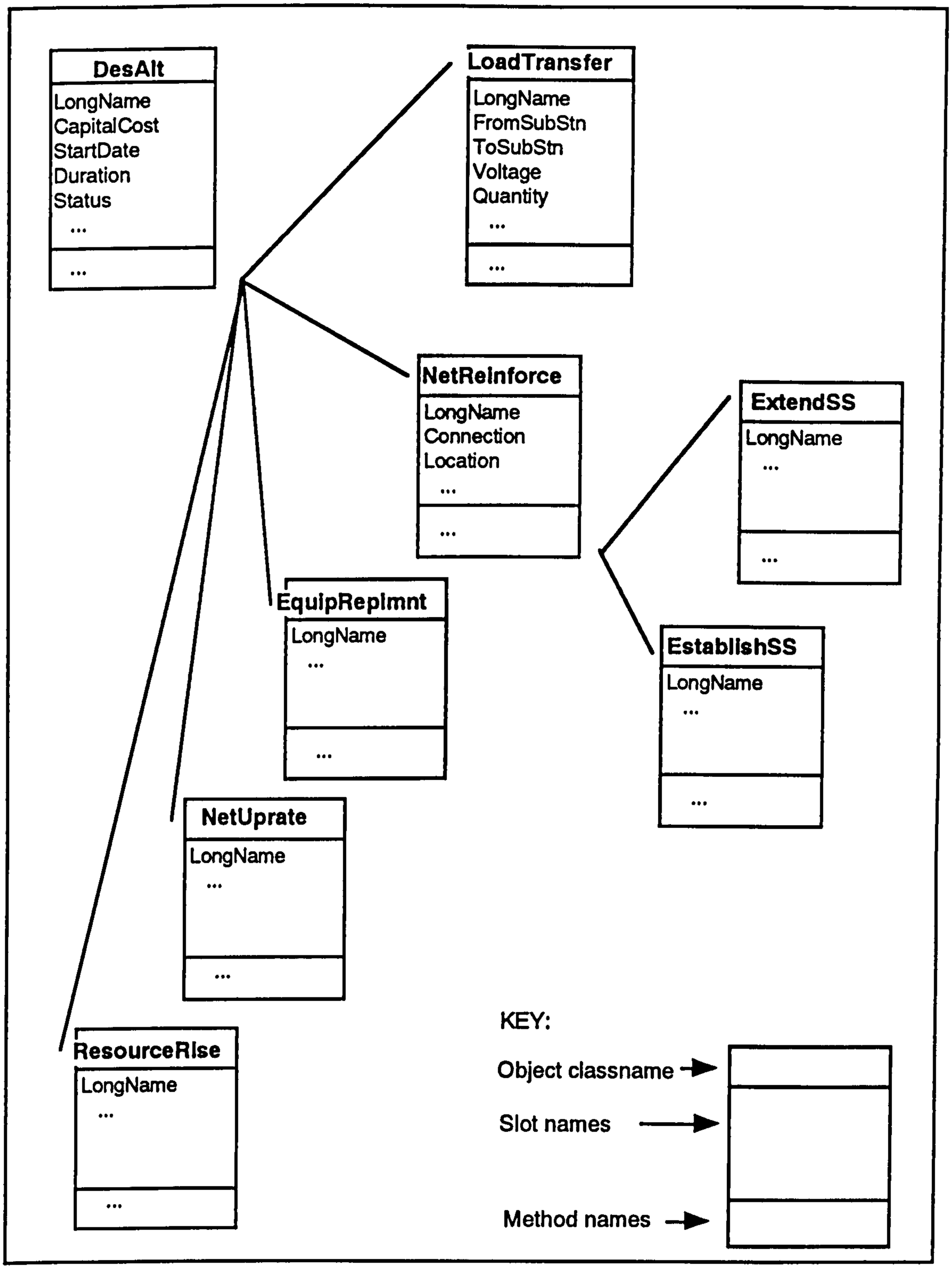


Figure8.2 Design alternatives (DesAlt) object hierarchy.

By representing "connection" and "location" aspects of "network reinforcement" (see figure 7.4) as slots in the object class "NetReinforce"

both the sub-classes "ExtendSS" and "EstablishSS" can inherit these facets of the network reinforcement class of design alternatives. In the figure limited detail of the representation of the design alternatives as a hierarchy of objects is shown so as not to distract from appreciation of the initial mapping from the SGN representation by cluttering the structure with detail. Further details of the representation of design alternatives are given below where the behaviour of the design alternatives object hierarchy is described in relation to other object hierarchies representing further aspects of the design situation.

During the process of designing using the design support system a designer will explore a number of design alternatives. Each time he considers a new approach an instance of the lowest level design alternative object class is created whose slots capture the designer's decisions. For example, if the designer considers two different load transfer possibilities and also the possibility of extending a sub-station three instances of objects will be created as shown in figure 8.3.

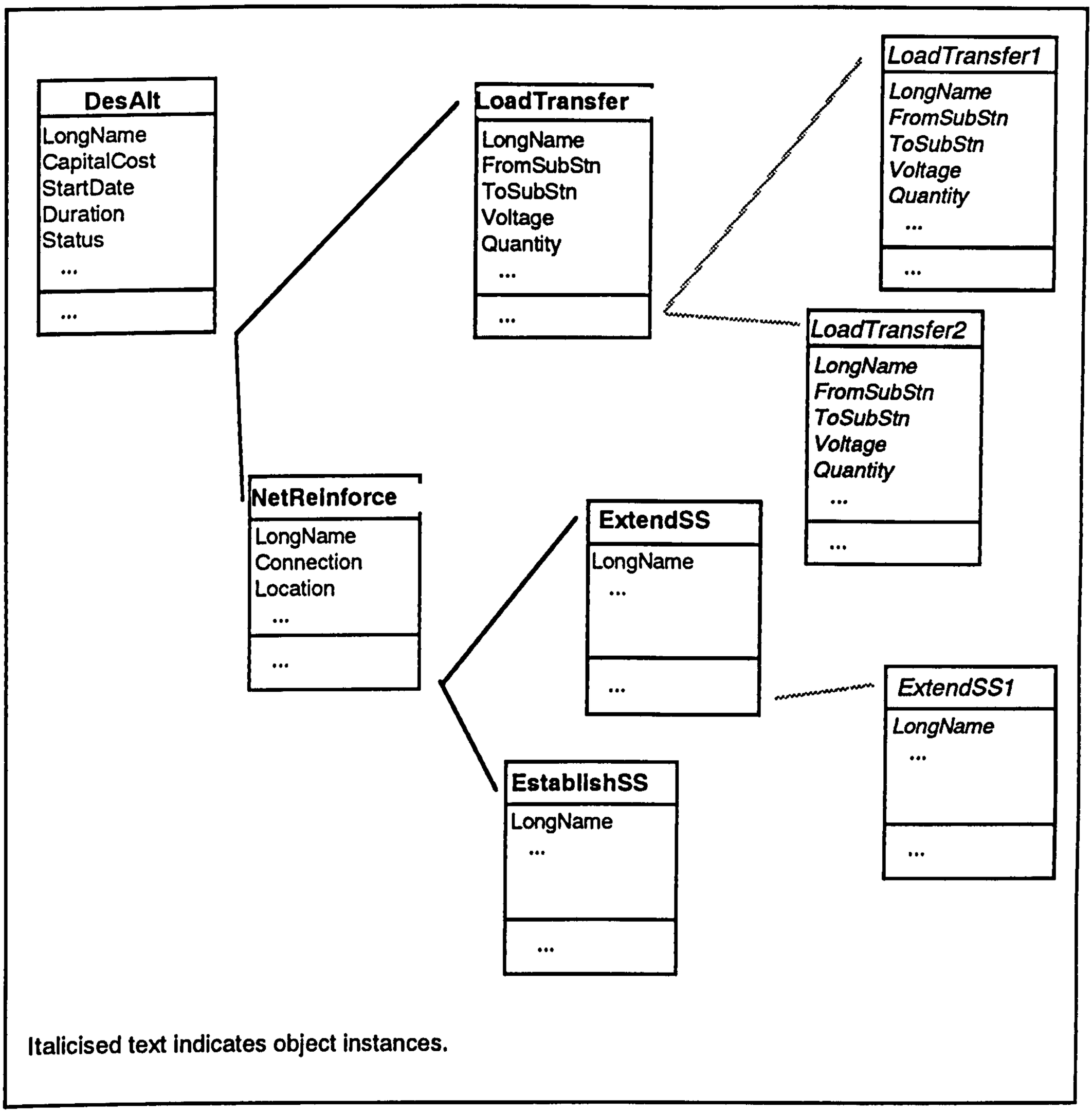


Figure8.3 Instances of design alternatives in relation to DesAlt object classes.

The design log (described later in this chapter) makes use of these representations to support the designer in combining a number of design alternative represented as lowest level object instances to assemble a complete design proposal.

2.1.2 Design commitments as an object hierarchy

The qualitative design commitments set out in the SGN of figure 7.5 of chapter 7 can be represented, in a manner similar to that of the design alternatives, in the form of an object hierarchy. In this case each object

class represents a type of design commitment. If the hierarchical relationships are thought of as a series of "is-a" links distinctions between the { and -[relationships in the SGN do not need to be carried over into the representation for this particular set of object classes once we consider the role of the design commitments in supporting the evaluation and justification of design alternatives.[1] Thus, the hierarchy shown in figure 8.4, the "QualComm" hierarchy, consists of classes of objects representing the qualitative design commitments. Each object class includes a slot which contains a description of each object, and the rationale for each design commitment which can legitimately be inherited by sub-classes since they represent specialisations of their super-class. The attribute (slot) named ArgumentSummary is used by the design log (described later).

[1]Meyer (op.cit. p.333) identifies client and inheritance relationships among objects. "Inheritance means 'is", client means 'has'. Inheritance is appropriate when every instance of B may also be viewed as an instance of A. The client relation is appropriate when every instance of B simply possesses one or more attributes of type A". In the QualComm hierarchy the inheritance case applies.

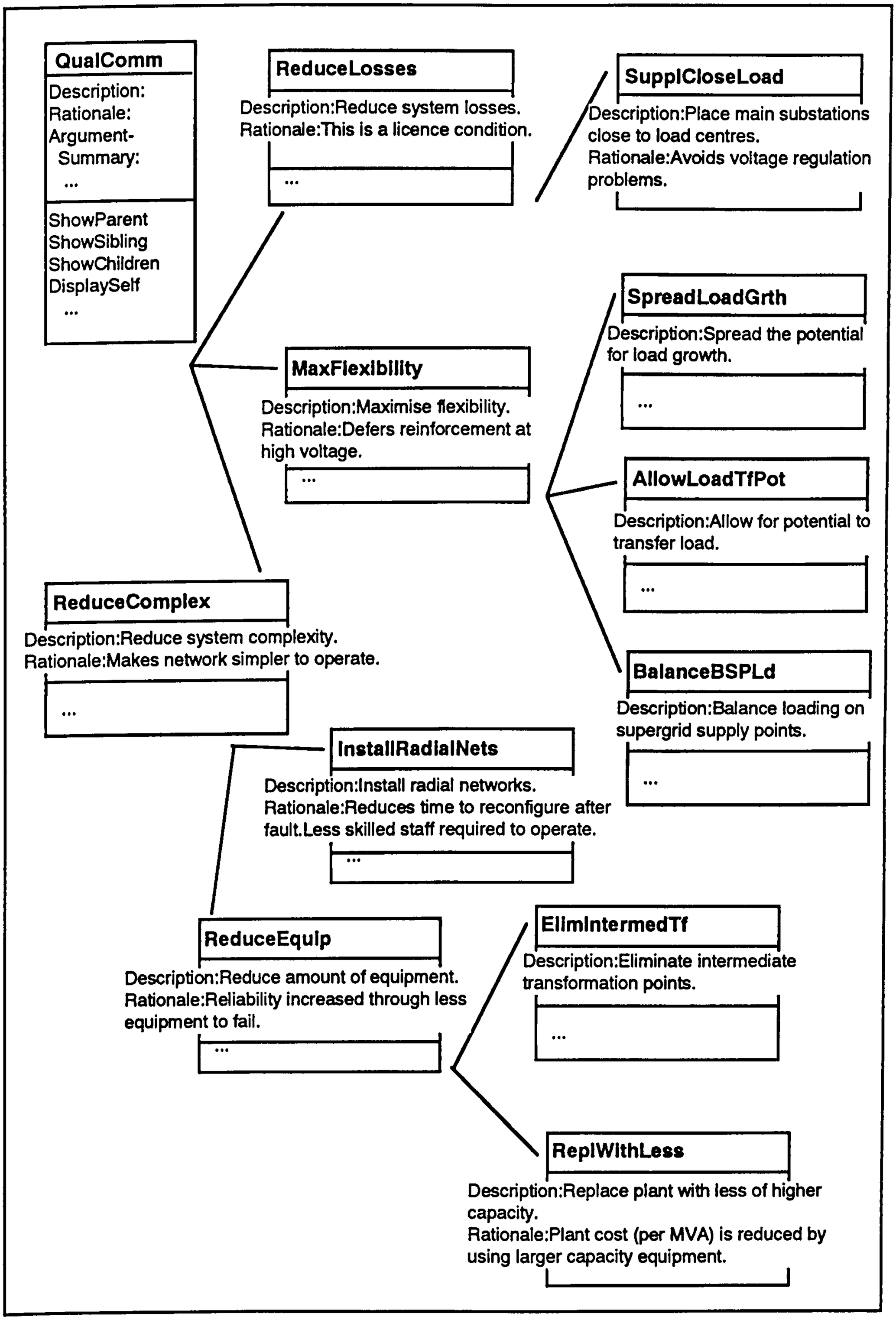


Figure 8.4 Design commitments (QualComm) object hierarchy.

During the process of designing using the design support system the designer has design commitments brought to his attention for consideration (he can choose whether or not to pursue them) depending on what sort of design alternative he is considering. The way in which this is achieved by message passing between objects in the DesAlt and QualComm object hierarchies is described in the next section (2.1.3). Once the designer has shown an interest in exploring the design commitments (by following up the system's prompting) he is able to explore the commitments which are more general, more specific or which are alternatives "at the same level" as the starting focus. Methods ShowParent, ShowChildren, and ShowSiblings which are defined in the QualComm object class and inherited by all sub-classes support each of these kinds of exploration respectively. An example of these methods in use is given in section 2.1.3 below. The DisplaySelf method, also inherited by all sub-classes, reveals to the designer a description of a design commitment and the rationale for it. Data to enable this is recorded in each object class (in the QualComm hierarchy) in the (attribute) slots Description and Rationale respectively.

2.1.3 Linking design alternatives to relevant design commitments using message passing

As designing proceeds, the designer chooses which design alternatives to pursue. The consideration of a design alternative prompts the system to make the designer aware of relevant design commitments. The designer will be interested in the design qualities that may be associated with a design alternative for a number of reasons (as we have seen in the constructed accounts of designing given in chapter 7). He may justify a design alternative by referring to the design qualities it embodies or he may evaluate and eliminate an alternative by reference to design qualities. He may use design qualities in a more subtle way to make the case for an alternative by comparing its design qualities with other alternatives that have been considered.

In the light of these findings, the association of design commitments with design alternatives is implemented in three ways, by linking design alternatives with design qualities which support the case for the design alternative (supporting qualities), by linking to those which counter-indicate its suitability (countering qualities), and finally by linking to qualities which could be generally relevant to the design alternative (relevant qualities). These links are achieved by message passing between objects in the DesAlt hierarchy and objects in the QualComm hierarchy. An example of this linking is shown in figure 8.5.

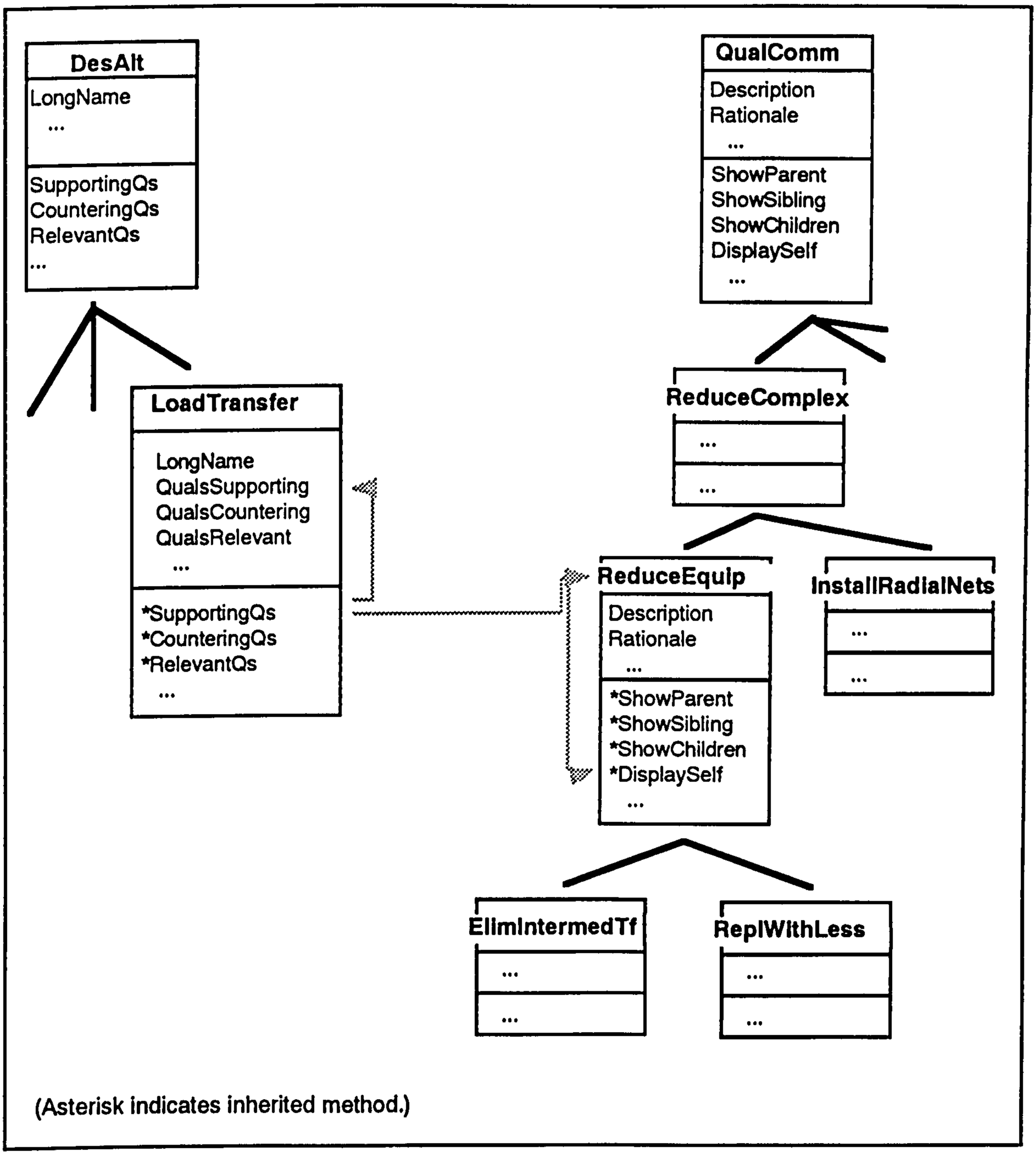


Figure 8.5 Link of design alternative to relevant design commitments using message passing.

The links between design alternatives (DesAlt) and design qualities (QualComm) is illustrated (with reference to figure 8.5) as follows. When the designer is considering the possibility of proposing a load transfer he is invited to look at the design qualities which support, counter-indicate or are otherwise relevant to load transfers. There are slots associated with the load transfer (LoadTransfer) class of design alternative (DesAlt) for each of these associations. The slots contain lists of the appropriate sub-classes of the design qualities (QualComm) object hierarchy. So, for example, if the designer elects to consider supporting qualities for the load transfer design

alternative, the method SupportingQs (inherited by LoadTransfer from DesAlt) will send a message to the object class(es) in the QualComm hierarchy which are listed in the QualsSupporting slot of LoadTransfer. The message instructs the receiving object(s), in this case ReduceEquip, to execute its method DisplaySelf (inherited from the QualComm object class). The dotted lines in figure 8.5 illustrate the message passing (leaving out the operation of the inheritance mechanism in both object hierarchies). This linking of the DesAlt and the QualComm hierarchies allows the designer's attention to be focused on design qualities (ReduceEquip in this example) which are relevant to what he is currently considering (here LoadTransfer).

The explicit classification of design qualities into a hierarchy can now be made use of, the designer can explore more general (ReduceComplex in this example), more specific (EliminateIntermedTf, ReplWithLess), and alternative (InstallRadialNet) design commitments based on the initial focus (ReduceEquip). Links from the design alternatives to countering and relevant qualities operate in the same way using methods CounteringQs and RelevantQs (inherited from DesAlt) and slots QualsCountering and QualsRelevant in the LoadTransfer object class in place of the slot QualsSupporting and the method SupportingQs. The exploration of design commitments open to the designer in the context of his consideration of a load transfer type of design alternative can be seen as a traversal of the QualComm hierarchy shown on the right hand side in figure 8.5 starting from the initial focus point of ReduceEquip.

2.1.4 Leaving Initiative with the Designer

The purpose and function of the interface which is provided to demonstrate the behaviour of the components implemented has been stated above in the introduction to this chapter. Even in this it can be seen that where the initiative lies (with the designer/user or the system) at any instant is not haphazardly determined. The support system should help the designer, this determines its role, as described above, in *focusing* on *relevant* design commitments. It has a similar role in *suggesting* an *appropriate* context for reviewing the electrical network which is described below (in section 2.2). The choice as to whether or not the links are activated (i.e. the suggestions are pursued) however rests with the designer. The system is required to make the designer aware of potentially relevant factors in a timely fashion as designing proceeds. By bringing supporting, counter indicating and otherwise relevant design qualities to the attention of the designer, the model which has been implemented in the design support system is rendered more visible, more open to inspection - and thus possibly open to dismissal or refutation by the designer.

Making the designer take the initiative is intended to invoke the feeling that responsibility is left with the designer. The designer directs exploration of the design alternatives, the system's role is to give him timely and relevant information to support his decision making whether it be trading off, evaluating alternatives, or building the case for justification of a

combination of design alternatives which will constitute a design proposal.

2.2 Representing the Design Context and Using the Design Activity to Focus the Context

Mapping of the systemic grammar network shown representing the electrical network (context) relevant to designing as figure 7.3 in chapter 7 and the way relevant links are made to it from the representation of design alternatives (the DesAlt object hierarchy already described) are described below.

2.2.1 Design context as an object hierarchy

One of the ways in which a designer can be assisted with his designing is by giving him access to information about the design context in a focused manner[2]. The designer's design activity, i.e. what aspects of the design problem or what approaches to its solution he is currently following, determines what, from the "surrounding" context, is currently relevant to what he is doing. Thus the *relevant* context is anything which will affect or be affected by the design alternative currently under consideration. In the case of the primary electricity distribution network designer the design context is represented as the electrical network (context) given in the systemic grammar network presented as figure 7.3.

Representing the electrical network context as an object class hierarchy requires that the sub-class to super-class links be viewed as being of two kinds. The object class hierarchy chosen to represent the electrical network context captured by the systemic grammar network of figure 7.3 is shown in figure 8.6.

---

[2]The designer is working within the design situation which has a number of aspects. The terms in which these aspects have been represented for the design situation studied here have been given in chapter 7 (see figure 7.1). The design context referred to here is restricted to the electricity distribution network expressed in the terms of the systemic grammar network of figure 7.3 in chapter 7.

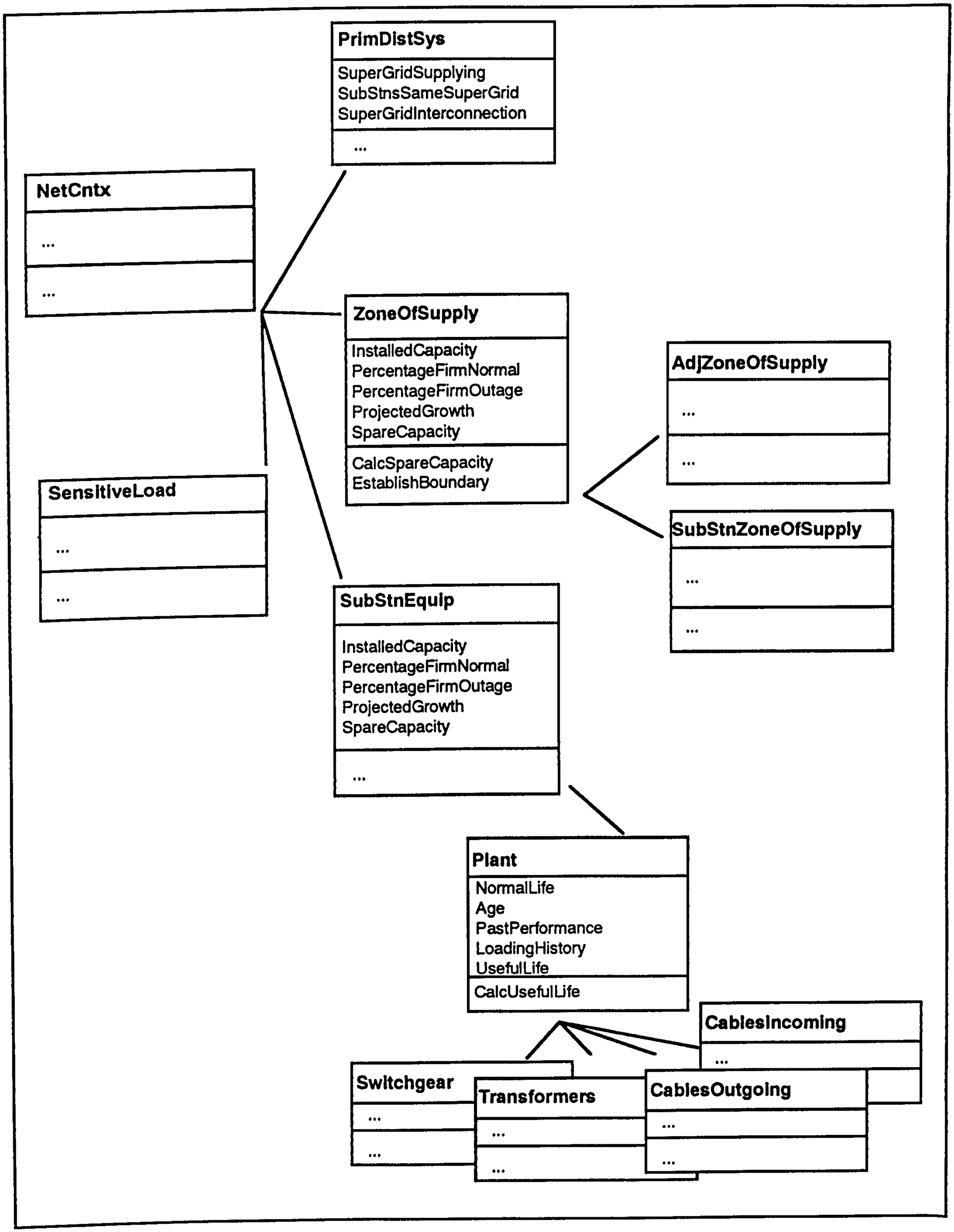


Figure8.6 Design context (NetCntx) object hierarchy.

The relationship between the object class NetCntx and the sub-classes ZoneOfSupply, PrimDistSys, SensitiveLoad, and SubStnEquipment is of the "is-part" type, whereas the relationship between these latter object classes and their subordinates is of the "is-a" type. The introduction of the object class ZoneOfSupply (cf.systemic grammar network shown in figure 7.3) with sub-classes SubStnZoneOfSupply and AdjZoneOfSupply permits the terms to the right of these (in figure 7.3) to be represented as slots and methods in the ZoneOfSupply object class which are consequentially associated with SubStnZoneOfSupply and AdjZoneOfSupply by inheritance. The term "spare capacity" in the context of the term "zone of supply" and terms to the right of it are represented as slots, and the term "bounded/delimited" and the terms of increased delicacy to its right are represented as a method (EstablishBoundary), in the object class ZoneOfSupply[3].

The term "spare capacity" associated with substation equipment can be treated in a similar way to the term "spare capacity" in the context of zone of supply i.e. by appearing as a series of slots associated with the SubStnEquipment class of objects. The introduction of an object class called Plant allows terms in the systemic grammar network to the right of switchgear, transformers, cables(in) and cables(out) which are common to all plant to be attributes inherited from Plant by each of the lowest level object classes which can then separately represent attributes which are special to each (terms of increased delicacy associated with switchgear, transformers, etc., are not actually shown in figure 7.3).

The terms to the right of "primary distribution system" in figure 7.3 are represented as slots associated with an object class named PrimDistSys. The relationship of "projected load growth" to "major future developments" and "planning load estimates" (same figure) represent the context within which reference to sources of information (external to the support system) is appropriate. A complete implementation of a design support system would at least provide a prompt to the designer suggesting reference to external information sources and give contextual information as a "search" parameter. In a more ambitious integrated design support environment, where different computer-based information sources were linked and co-ordinated by the design support system, rules and methods

---

[3]Establishing the boundary is a procedure which takes account of the boundary defining terms (from the systemic grammar network). To establish a boundary in a particular situation would require access to a database of electrical network details (about sub-stations and their interconnections) consequently the method EstablishBoundary has not been implemented. Stated briefly, it would be realised as a method in the ZoneOfSupply object class which, when passed a parameter identifying the sub-station under consideration, would "filter" the network database by applying rules associated with each of the aspects for establishing the boundary given in figure 7.3 (terms to the right of the term "bounded/delimited"). The resulting "subset" of the electrical network would then be presented to the designer in graphical form for his consideration.

would allow the designer to view data (for example planning load estimates) via rules making appropriately focused links (like those for retrieving portions of the electrical network through use of the EstablishBoundary method described above). An integrated support environment in which heterogeneous design resources are co-ordinated and linked by a knowledge based design support "core" is outlined in section 3.1 below.

2.2.2 Using Designer's Interest to Suggest Context

The NetCntx object hierarchy shown in figure 8.6 supports the representation of the design context in terms relevant to the designer. When he is designing, specifically when he is developing or considering a design alternative, his interest in the electrical network context will be restricted (or focused) in two ways. Firstly, what he is considering will determine the *aspects* of the electrical network context he is interested in. For example, if he is considering transferring load between sub-stations he will be interested in the loading on parts of the network (e.g. the spare capacity in a zone of supply, projected load growths, and similar sorts of aspects of the network); on the other hand if he is considering extending a sub-station by adding new plant he will be interested in the existing plant (switchgear, transformers, and cable connections) at that sub-station. This focusing on particular aspects of the network context will be termed the abstract focus or abstract context.

The second way in which his interest in the electrical network is focused is dictated by the particular (geographical) part of the network where the design proposal is to take effect, that is the particular part of the network, the actual substations for example, which are to be affected by the load transfers or new plant installation. This aspect of focusing will be termed the concrete focus. The concrete focus or concrete context is indicated by the designer when he specifies which sub-stations are the subject of a design alternative (which sub-stations are affected by it). For example, in the case of a load transfer, the designer will indicate which sub-stations are to supply and receive the transferred load, whereas in the case of sub-station extension, he will name the sub-station which is to be extended. This provides information which can be used to determine the concrete network context relevant to the designing taking place. In the design support system, the designer's design decisions (what design alternative he is considering) can be used to suggest a focus of reference to the (network) context. As the designer's interest shifts, so the network context which is relevant also shifts.

Section 2.1.3 above describes a way in which design alternatives represented as an object hierarchy can be linked to relevant design commitments, similarly represented, using message passing between the two object hierarchies. Sections 2.2.2.1 and 2.2.2.2 below describe how by a similar mechanism abstract and concrete focusing on the network context can be achieved using message passing from the design alternatives object hierarchy (DesAlt) to the network context object hierarchy (NetCntx).

2.2.2.1 Abstract focus using message passing

The initiative for examining the electrical network context during designing is left with the designer (for the reasons given previously). However, should the designer elect to turn his attention to the network (context) while he is considering a particular design alternative, the design support system should be able to help him by suggesting a focus of attention based on what designing he has been doing. So that the initiative for looking at the network (context) at all, and for choosing the focus in it, will rest with the designer, the links between design alternatives (the DesAlt object hierarchy) and the network context (NetCntx object hierarchy) are represented at two commitment levels which represent the suggestion of a focus (establishing a link between the two hierarchies and presenting the suggested focus to the designer) and the triggering of a focus (actually transferring attention from the designing of an alternative to the consideration of the electrical network). The latter being effected when the designer chooses to follow up the suggested focus. Slots named NetCntxAbstractSuggestion and NetCntxAbstractTrigger are defined in the DesAlt object class for inheritance by all sub-classes of objects in the design alternatives object hierarchy. These slots are used to refer to object classes in the NetCntx object hierarchy and as such constitute links to suggested or triggered foci.

The method DispNetCntxAbstractSuggestion defined in the DesAlt object class is used to display the suggested focus (NetCntxAbstractSuggestion) to the designer whilst the method TriggerNetCntxAbstract, also defined in the DesAlt object class actually shifts focus to the network context suggested (using the NetCntx sub-class recorded in the NetCntxAbstractTrigger slot) if the designer elects to follow up the focus suggested. Figure 8.7 shows the links between design alternatives and the network context which support focusing attention on aspects of the network (context) relevant for designing load transfers.

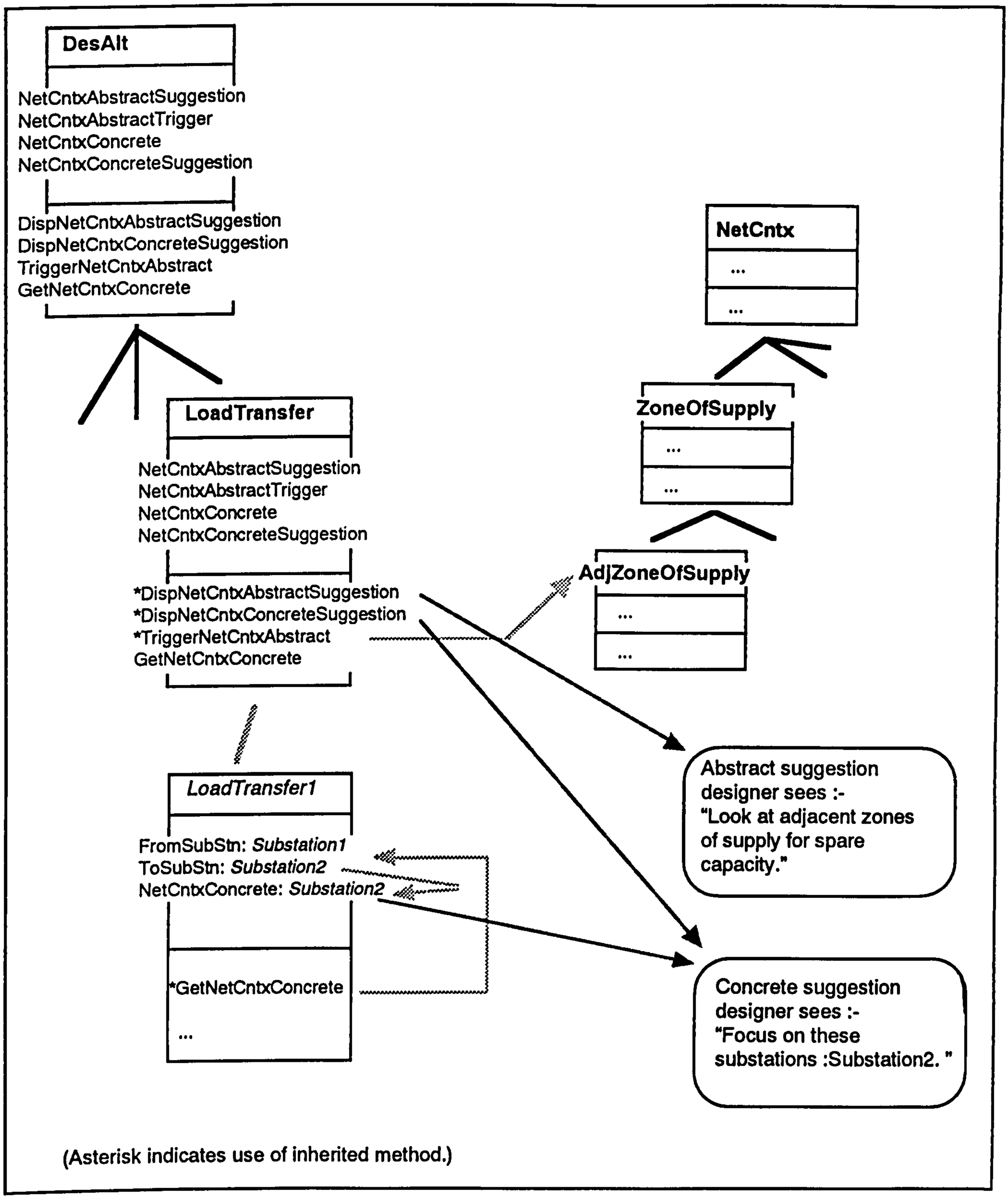


Figure 8.7 Linking a design alternative to the electrical network context.

2.2.2.2 Concrete focus using message passing

The abstract focusing mechanism described above identifies properties of the network which are likely to be of interest to the designer on the basis of the design alternative he is currently considering. The concrete focus operates in a similar manner but in this case an attempt must be made to detect which specific parts of the network (which particular sub-station, circuits, plant, etc.) are relevant. Two methods and two slots are defined in the DesAlt object class to effect associations between the DesAlt hierarchy and the NetCntx hierarchy. The slots are NetCntxConcreteSuggestion and NetCntxConcrete. Both slots are used by method DispNetCntxConcreteSuggestion to present a suggested (sub-station) focus for examining the network context. A further method, GetNetCntxConcrete, is invoked when the value of NetCntxConcrete is required. This method determines which sub-station(s) should be focused upon by referring to the data the designer has entered about a design alternative.

Whilst each DesAlt sub-class contains a method called GetNetCntxConcrete each method differs in the way it obtains the context according to the object class to which it belongs. For example, in the case of load transfers the sub-station receiving the transferred load (the value of the slot ToSubStn in an instance of a LoadTransfer object) gives the concrete focus; in the case of a substation extension (object class ExtendSS) the value of the slot SubStnName gives the concrete focus. Figure 8.7 shows how the concrete focus to be suggested to the designer is obtained when the designer has most recently been considering load transfer as a design alternative.

2.2.3 Leaving initiative with the designer

The design support system should allow the designer to transfer his attention, whenever he wishes to do so, from actually specifying a design alternative to the perusal of information pertinent to its development. Here, this means that the designer needs to be free to review the network context relevant to his design activity. When it is possible for the design support system to do so it should help the designer by suggesting a focus for network review in terms of which properties of the network and which specific parts of it are likely to be of interest. The designer is always free to ignore or override the suggested focus. The links between design alternatives and the network context described above are capable of providing focus suggestions to the designer. As with links between design alternatives and design commitments, the choice of whether links are pursued rests with the designer. The system's role is to make the designer aware of potentially relevant material when it is appropriate to do so.

2.3 Supporting Review and Comparison of Design Alternatives with a View to Making a Justifiable Design Recommendation

The implementation of a rudimentary design log to support reviewing and comparing design alternatives with a view to making a justifiable design recommendation is described in this section.

2.3.1 Design log as an object class

The design log is implemented as an object class. Design logs for different design sessions for different design situations could be easily stored and recalled by making each one an instance of the design log object class although this facility has not been implemented in the work presented here.

The slots of the design log are used to record class-based and chronological lists of design alternatives pursued (i.e. instances of design alternatives created) during designing with the design support system. The main power and extensibility of the design log lies, however, with the methods it contains. When a designer considers a design alternative, as has already been shown (figure 8.3), an instance of a design alternative object class is created. The methods associated with the design log object class use these instances of design alternatives as a resource to assemble information about a design alternative and to present this to the designer.

Two simple methods, SummarizeAllAlts and SummarizeClassOfAlts, use the chronological list of design alternatives considered, a list of alternatives grouped into object classes, and the instances of design alternatives themselves to present summary information to the designer either about all design alternatives considered so far and all design alternatives considered so far of a particular class, e.g all load transfers, respectively.

2.3.2 Associating design commitments and network contextual information with a design alternative

A designer may wish to review the case for a particular design alternative, to compare it with other alternatives, or to combine alternatives together to produce a substantiated design proposal. To show how a design log might assist in this a method, ShowArgumentforaDesAlt has been implemented. On request from the designer (who will specify which alternative he is interested in) this method accesses the appropriate design alternative object instance and provides the following:

- summary information about the alternative (selected slot values);
- an argument summary (value of slot ArgumentSummary)

associated with the design commitment(s) which support, counter-indicate, or are otherwise relevant to the design alternative (using the links described between the DesAlt object hierarchy and the QualComm object hierarchy described above in section 2.1.3);

- the abstract and concrete electrical network context information associated with the design alternative object instance (using the links between the DesAlt object hierarchy and the NetCntx object hierarchy described above in section 2.2.2).

Figure 8.8 shows the effect of activating the method ShowArgumentforaDesAlt following a request from a designer interested in an instance of a design alternative of the load transfer class.

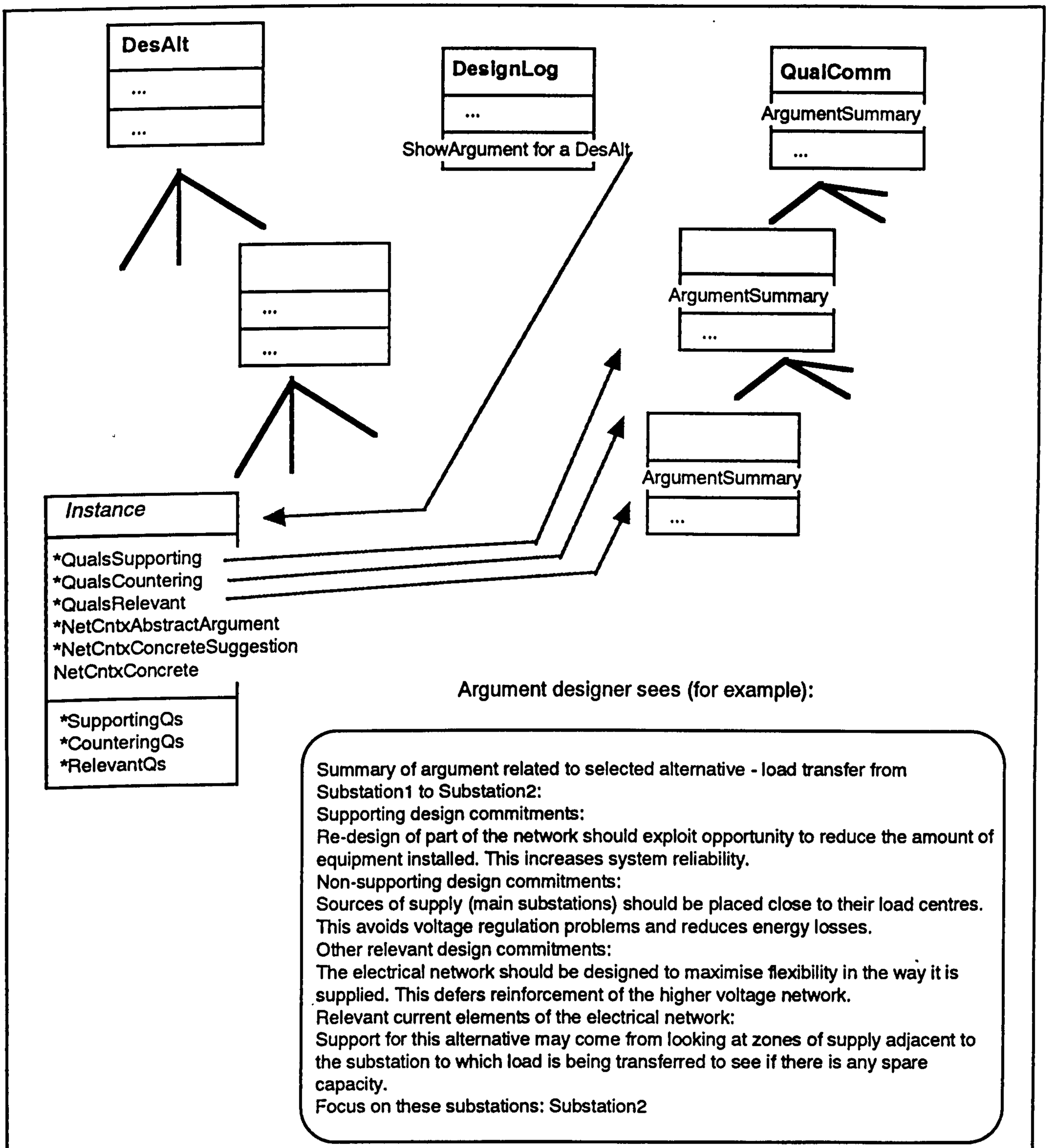


Figure 8.8 Showing the argument for a design alternative.

The method ShowArgumentforaDesAlt is a fairly modest illustration of the contribution design log methods might make towards supporting the designer in comparing design alternatives and composing a justifiable design proposal. However, two aspects of the illustration are important for more ambitious design log services. Firstly, the design log functionality is achieved through methods associated with the design log object class. Additional functionality and behavioural complexity can therefore be achieved in an entirely modular, autonomous fashion without loss of control over system complexity or loss of perspicacity. Secondly, the design log uses *instances* of design alternatives as the primary resource for the services it provides. As it is implemented here (i.e. in the method ShowArgumentforaDesAlt) the supporting design commitments being associated with a design alternative are essentially abstract - in the sense that the same qualities (commitments) will be retrieved for all instances of a particular design alternative object class. However, this need not be the case.

It is envisaged that a designer would make reference to supporting, counter-indicated and other relevant design commitments in the course of designing an alternative. He could easily be provided with the means to select some or all of these commitments and to customize (particularise) them - by adding his own comments for example. This activity could easily be recorded by trivial extension of what has been implemented here (effectively by recording what the designer selects or enters in (slots) QualsSupporting, QualsCountering and QualsRelevant to override the defaults inherited from the class of design alternatives to which the design alternative instance belongs). The design log makes its references (addresses its messages to) the instances of design alternatives and would therefore "pick up" this particularisation automatically even if ShowArgumentforaDesAlt were to be left in its current rather unsophisticated state.

3 Relationship of Support Described to Broader Context of Electricity Distribution System Designing

The design situation representation described in section 5 of chapter 7 offers an organizing framework not only for some self-contained knowledge based design support system but (more importantly) for a design management system that integrates heterogeneous design support resources. The resources presently available to a designer of electricity distribution networks are both formal and informal, computer-based and non-computer based. A system capable of assisting the designer with managing his design activity as a whole should be able to both suggest what it would be useful to focus attention upon and support the designer in considering a suitable range of relevant issues. A knowledge based design support system based on the design situation representation developed here could initiate access to a range of design resources (e.g. databases of plant, equipment and circuit details and network analysis programs) and "filter" the responses from these resources so that the designer is presented with a

focused set of information relevant to what he is doing.

At present, if the designer wishes to make use of plant and circuit databases he must select what is relevant either (at best) by posing a query himself or (worse) he must wade through general purpose lists and reports, and other diverse "raw" information sources to find what is relevant. Similarly, many of the analysis programs for load flow and security assessment studies require large volumes of specially formatted data to be prepared and input and they produce correspondingly large volumes of numerical analysis data as output, from which the designer must select what is relevant. A design support system that can mediate between (say) analysis programs and databases describing network components and their interconnections which could retrieve relevantly focused data from them, structured in accordance with what the designer's (current) interests are would be a powerful design aid. Figure 8.9 shows, schematically, an example of the role a knowledge based design support system might play in co-ordinating the design resources which have been mentioned here and earlier in this chapter.

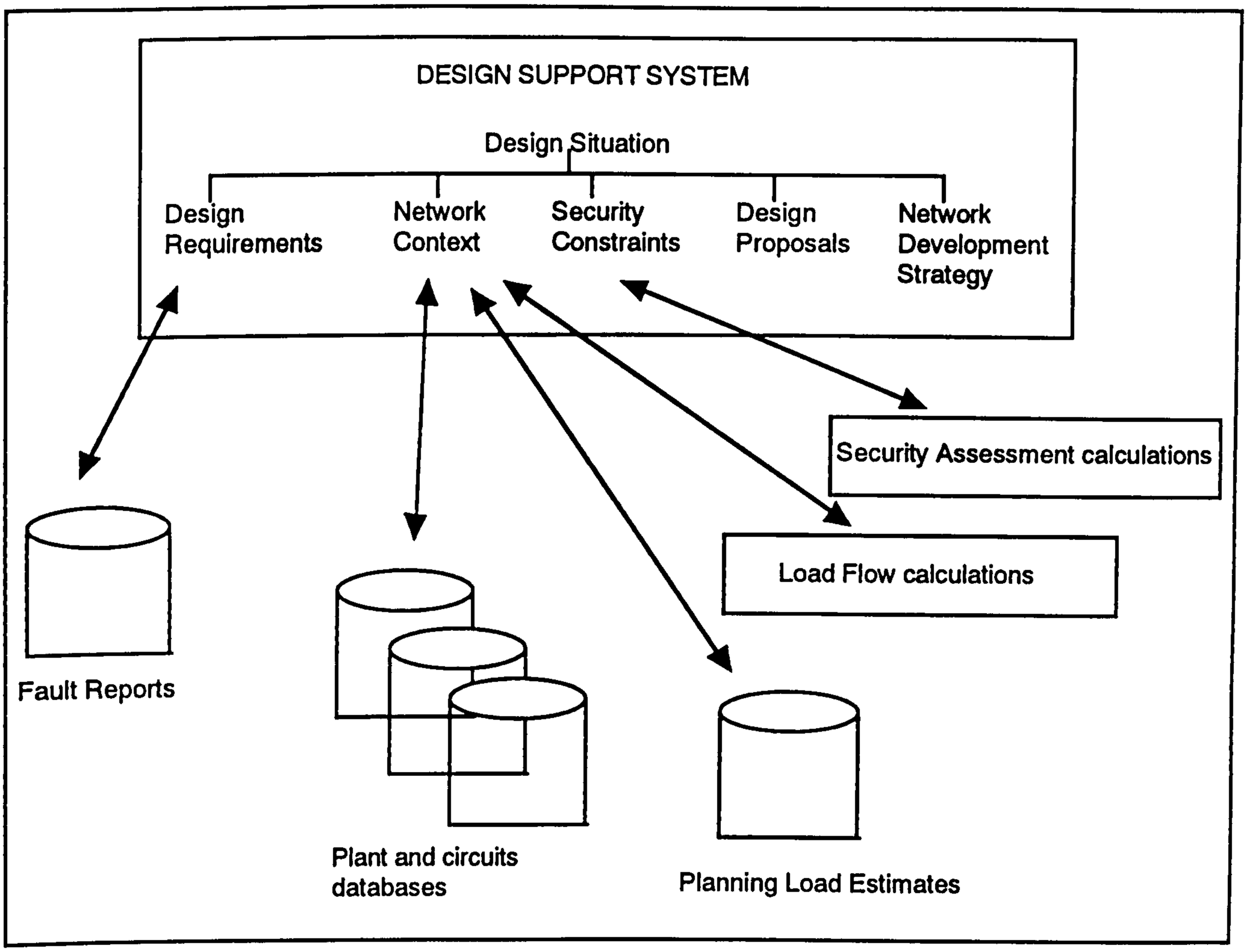


Figure 8.9 A knowledge based design support system managing some design resources.

4 Role of Knowledge Representation in Preserving and Passing on Design Knowledge

During the 1960s and 1970s there was a large expansion in electricity distribution and use (in the U.K.). Many of the more senior design engineers who now work in primary electricity distribution planning and design have decades of experience in the industry and were themselves actively involved in installing, commissioning and operating the distribution network back in the 1960s. Experience of the kind that arises from the quantity and variety of exposure to different work situations was relatively easy to come by in those days. Competent engineers were scarce and opportunities for novice engineers to take on responsibility and to learn from experience were plentiful.

The situation is very different today. In many urban areas saturation point has been reached in the sense that scope for large scale new

installations are limited compared with the ubiquitous development of twenty to thirty years ago. There are, of course, still demands for network expansion and the design work associated with it (London's Docklands development, until relatively recently, was a good example), however design work associated with major new developments tends to be given to those who are already experienced designers. Opportunities to acquire experience in the "traditional" way, i.e. by being given lots of different design problems of increasing complexity, is no longer the norm. Consequently, experts in primary electricity network design who retire today take with them a lifetime's expertise which is not easily replaced since the experiences which led to their expertise cannot now be obtained so readily. Primary electricity distribution network design is not the only area of electricity supply that is affected in this way. For example, for several years work has been reported on the incorporation of intelligent data handling into the control centres of the electricity generating and distributing companies[4]. One of the stated motivations for this is to capture and retain expertise in handling network control problems. The argument given is that as older, experienced control engineers retire, they are being replaced by less experienced engineers who, because of the increased reliability and stability of power systems inevitably get fewer opportunities to get first hand experience in controlling problem situations. The consequences here of inexperience are exacerbated by the increased complexity of the power networks which have to be controlled.

It can be argued, therefore, that there is a role for representing professional expertise simply to record and preserve it, and if suitably represented, to pass it on to less experienced practitioners. Thus, representing knowledge in some non-destructable medium may become part of a radical alternative means for assisting novices to become competent in a field more rapidly from fewer primary experiences of exercising their professional skills. There is no intention to imply that a designer can learn to design from a knowledge based system to support design. The design support system has a more subtle role - in supporting reflective practice - in assisting a designer to develop and broaden the ways in which he views a design situation. In such a role, the design support system's ability to reveal the basis for its support and to provide an interactional style which encourages the designer/user to play an active role in assessing the appropriateness of the design suggestions other support provided would be crucial to success.

The activity of knowledge elicitation has an important role to play, per se, in promoting conversation about designing at a valuable, but rarely naturally occurring, level. During one of the interview sessions associated with this case study a trainee engineer close to the end of his apprenticeship was present as an non-participative observer (listener). At the end of the interview session the trainee was invited to comment on his

---

[4] For example in the published proceedings of the four symposia which have been held on expert systems application to power systems, held in Stockholm and Helsinki, 1988; Seattle, 1989; Tokyo and Kobe, Japan, 1991, and Sydney, Australia, 1993.

experience of sitting in on the session. He indicated that he found the session "fascinating" and although he had been "training" with the design engineers for some time (several months) he had "never heard anyone talk about the job that way before". Reported accounts of knowledge elicitation exercises often include observations, usually in the form of incidental remarks, about the value and insight into their own expertise that an expert subject gains from the elicitation activity itself.

Knowledge elicitation techniques, applied in conjunction with a suitable representation formalism, have been used, on occasion, to establish and formalize the expertise of a group of experts as an end in itself (rather than as a step in a process towards implementing a knowledge based system). The reflection on their practices prompted by knowledge elicitation exercises changes experts' perceptions of those practices and hence can be of value regardless of any application of the outcome to realize a design support product. This section concludes with one further issue which although highlighted by the particular environment of the case study and described here, in terms of it, is of wider interest.

This last matter relates to preservation of a value system (Vickers, 1968) which if it remains only as a phenomenon exhibited by a group of professionals, is vulnerable to changes in the working environment i.e. if it ceases to be reinforced and reinterpreted through the working practices of the group. The design commitments elicited and represented in the case study presented here represent the realization of a value system as it applies to, or affects, a particular professional practice. A value system evolves gradually, there are shifts in emphasis and changes over time but in a relatively stable working environment the value system ensures stability, a constancy of "larger" purpose over time, giving the overall evolution (here of an electrical distribution network) a valuable coherency. A mature value system is subtle and multi-dimensional.

A value system shared by professional designers in a particular field both influences what the designers design and at the same time is created and maintained by what the designers design. If a major perturbation is introduced from outside, one which constrains the designers to behave differently from the way which accords with their professional value system, then the value system will disappear because it will no longer be embodied in the designers' activity. If the shift to a new value system is imposed and maintained for some time designs will cease to manifest the usurped value system; the original value system will cease to play a part in evaluation of the designed objects. Eventually the designers themselves will naturally be replaced by designers whose only experience will be of designing in the sense it has acquired and the meaning it has come to have through the new value system.

Recent radical changes in how electricity is generated and distributed have led to a discontinuity in design practice as a result of a radical value system shift. The design commitments elicited and represented in this case study are a distillation of forty years of experience of a community of

professionals. Whether or not the value shift is deemed a good or bad thing is not the issue here. The outcome is that the ability to design in the way the designer who is the subject of the case study designs will inevitably be "lost" within a few years because the ability to appreciate and effect designs in accord with the qualities consequential upon the value system which he uses will cease to be exercised. The value system only exists through its use. Thus, the representation of design expertise recorded in the chapters of part 2 of this thesis may eventually have their greatest value as some curiosity in a museum of knowledge.

CHAPTER 9

Concluding Remarks

"Two roads diverged in a wood, and I -
I took the one less travelled by,
And that has made all the difference."
Robert Frost (The Road Not Taken)

This thesis has investigated how the capabilities of second generation knowledge based systems technology can be brought to bear on the task of supporting engineering design; the powers and limitations of models of design competence as the basis of knowledge based design support have been taken into account; the work has been firmly set in the context of the importance and effect that an understanding of the nature of design as a human social activity has on such a study.

1 Summary

The contributions of variously motivated studies of the nature of design as a human activity have been investigated to see how designing can be understood in the light of these studies each of which contributes different emphases. Three strongly interconnected aspects of designer's competence have been analyzed, namely, the abilities to make trade-offs, to generate and evaluate alternative designs, and to justify design decisions. A review of the spectrum of applications of knowledge based approaches to design support has been presented; problems and limitations which have been experienced with the various approaches have been identified by focusing on the extent to which they handle distinguishing characteristics of design competence. The problems with knowledge based support of design have been analyzed to identify required components of a model of design competence on which to base a design support system. The Competent Systems architecture has been proposed as a framework for supporting designer's activities.

The inseparability of competent behaviour and the professional setting in which it takes place has been discussed and related to methodological issues which in turn direct knowledge elicitation activities. Requirements to be met in supporting design practice have been explored and discussed. The implications for using knowledge based systems technology to support design of the understanding that a designer, in practising his profession, must account for and take responsibility for his decisions; that he will consciously use reflection to be able to cope with new situations and learn

from his experiences; and that his work will be defined and framed in terms of the standards and norms of a professional setting have been set out. These implications have been related to the consequential properties, behaviour, and structure they necessitate in the components of a knowledge based system for design support.

An empirical study of a designer's practice has been carried out to develop and test a representation of a design situation adequate for designing competently. An appropriate variety of knowledge elicitation, knowledge analysis, and knowledge representation techniques have been applied in the practical study. The way these have been used and their value in eliciting a representation of the design situation has been explained and their setting within the coherent methodological framework consequential upon the expressed view of what constitutes competence has been demonstrated.

The elicited design knowledge from the case study has been shown to be interpretable in a way consistent with the general understanding of the nature of designing and of design practice which has been presented. The proposals for effectively supporting design using knowledge based systems technology have been illustrated using the case study domain. The feasibility and effect of implementing some of the essential components of a knowledge based system to support design by mapping them onto a suitable software platform have been demonstrated.

Lawson has suggested that one of the major roles of a professional designer tutoring a student in the discipline is "to move the student around from one part of the problem to another" and the job of the design student is "to learn to do it for himself" (Lawson, op.cit., p.81). Part of the strategic role of a design support system is to help the designer to manage his designing, to make it easier for him to move around from one part of the problem to another, and to bring to hand what is relevant to the designer's concerns at a particular point in the design process. A role closely connected to this is that of supporting the designer's reflection on the design situation and his own efforts within it to respond to what he sees as the design task. This thesis has identified the requirements for supporting design practice in this way and has specified components of a knowledge based system architecture which begin to meet these requirements. It has also been shown how these components can be realized in software implemented in a knowledge engineering environment based on the object-oriented paradigm. The particular suitability of software developed in accordance with the object-oriented paradigm for the implementation has been discussed.

To produce a valid and useful model of design competence it is necessary to study what designers actually do and to use knowledge elicitation methods and tools which are capable of supporting the elicitation of knowledge through meaningful conversation in Pask's sense and of representing the outcome. Through such conversations, supported by techniques such as teachback based on systemic grammar network

representations, the knowledge engineer comes to a shared understanding with the expert designer which is adequate for producing a model of the designer's competence which can be validated.

2 Conclusions

Whilst research in design theory has not produced a simple, easily representable, or in any sense definitive account of what the nature of designing is, nevertheless, there is a coherence in the body of work which has been undertaken. This should be taken into account by anyone attempting to support design, doing so makes a difference both to what is attempted and to what is produced in the way of design support systems.

The large number of applications, and the diversity of approaches in using knowledge based systems technology to support design can usefully be presented, compared and evaluated by identifying the view of design embodied in each of the systems which have been developed and the scope of support each provides. Their strengths and inherent limitations can be analyzed by establishing the extent to which each is able to accommodate a range of the salient aspects of designers' behaviour. This sort of analysis brings out issues which need to be addressed if the support given to designers by systems based on knowledge based systems technology is to be improved or extended.

Many of the limitations of knowledge based design systems can be seen to result from either over-constraining the design systems on the basis of an over-simplified view of what the design task entails or , in contrast, under-specification and representation in the design systems of what designing entails, resulting in an inability to give directed support. A thorough understanding of what is possible in terms of modelling expertise can usefully be brought to bear on tackling these limitations. Basing the design of knowledge based design support systems on models of competence gives an organizing framework for the components of the support system. Models of competence make the rich conceptual structures which constitute competence explicit. Design competence, in particular, consists of a rich variety of strongly inter-connected aspects of design behaviour, hence a model to represent it must be correspondingly complex.

Competent designers are capable of critically appraising their designs as they develop them; they make use of what they have discovered in evolving the design, both to advance it further, and to evaluate it. There are strong links between the generation of alternative designs and the making of trade-offs, and similarly the notions of critically appraising the alternatives as they are developed and of justifying the design which is finally proposed are strongly connected with one another. A knowledge based system based on a model of competence provides the basis for a design support system that takes these matters into account because it supports explicit representation of task structure, strategies and strategic choices and can therefore support the quality of interaction needed to give relevant, supportive assistance to a designer in a way that makes sense from the designer's perspective.

The concept of professional competence cannot be separated from the organizational setting within which it is exercised. Professional choices take place in a context of professional practice conforming to professional norms. What a competent designer chooses to do, therefore, is only intelligible within the normative context, as are the explanations and justifications he gives for the decision he makes. To be effective, any design support system must fit into the professional environment which sets the context for a designer's work, consequently, the relationship between design competence and professional design practice is of primary importance in investigations of how designers can be supported.

Basing a knowledge based design support system on a model of competence not only determines a suitable structure for the resulting system but also has methodological consequences for how the knowledge to be represented is elicited. A variety of knowledge elicitation methods and knowledge representations can be applied to the gathering and analysis of data and provided they are applied on an adequate methodological basis a competence model can be constructed. Pask's conversation theory provides a suitable paradigm in this respect. For a particular design domain, plausible accounts of designing can be constructed and checked for acceptability with the designer on the basis of knowledge elicited in a suitable way. Insight into the nature of designing, as it is understood more generally, helps in constructing these accounts. These accounts, elicited representations of the designer's appreciation of the design process and the designed artefact, and systemic grammar network representation of the design situation provide complementary representations of the designer's task and the context relevant to it which are useful for designing a knowledge based design support system.

A systemic grammar network can define a grammar which captures the representable aspects of a design situation relevant for competent designing, the generative capacity of the grammar can be used to check the adequacy of the representation prior to any implementation of a knowledge based system in software. A systemic grammar network representation of a design situation specifies the high level components of a knowledge based system. A general purpose knowledge engineering environment based on the object-oriented paradigm offers a suitably rich and flexible software implementation platform on which to map the architectural components necessary for supporting the designer. The salient characteristics of the object-oriented paradigm, the sorts of approaches to system development it supports, and the salient qualities of software developed according to its precepts, combine to render it particularly appropriate for implementing knowledge based systems based on competence models. The structure of the system which results from this process is one which has the potential for taking an organizing , co-ordinating, strategic role at the core of a design support system which unifies a heterogeneous collection of resources for presentation to a designer.

3 Further Work

There is a huge potential for further work in the area covered by this thesis. Two classes of further work are mentioned here. The first is work on sharing and evolving the ideas and issues which have been the concern of this thesis by communicating the work - sharing the perspective taken and developing it further - through discussing it with other workers in the same area who have different perspectives which might enhance and be enhanced by the one presented here. The second class of further work is more conventional in that it consists of some suggestions about how some aspects of work presented in this thesis might be developed further.

It would be of value to enter into a dialogue with other researchers attempting to support engineering design using knowledge based systems technology. Unfortunately such dialogues are difficult in this field since common terms and agreement about (even abstract) components of knowledge based systems architectures do not exist. It should be clear from the theoretical arguments of this thesis and the example of empirical study given as Part 2 that piecemeal comparison e.g of a knowledge elicitation aid or a knowledge representation formalism, or an implementation approach would not lead to any valuable outcome because (for example) how knowledge is elicited determines whether it can form the basis of a model of competence.

It might be considered useful, for example, to see the extent to which the decompositions of design knowledge implemented in reported work describing knowledge based systems for different engineering design domains can be viewed as corresponding with one another. Unfortunately, this sort of post-data gathering comparison is inherently limited in value since the methodological basis for the knowledge elicitation determines what can be elicited and what its status is as a basis for determining the components of a knowledge based system.

An example illustrates the problem. Doheny and Monaghan describe an expert system for the preliminary stages of (conceptual) design of building energy systems, IDABES, (Doheny,1993). The stage (preliminary) and level (conceptual) of design that they report corresponds to that of the case study used for this thesis although, clearly, the design domains differ. They identify the knowledge required for designing by distinguishing five knowledge components termed domain knowledge, constraint knowledge, procedural knowledge, analysis algorithms and solution knowledge. A superficial analysis of their description shows some correspondences with the main aspects of the design situation identified in the case study (described in section 5.2 of chapter 7). In IDABES procedural knowledge can loosely be associated with what has been termed here the task structure, constraint knowledge in IDABES includes both design requirements and heuristics about constraints between components, so these might be compared in some way with the design requirements and the quantitative design commitments (constraints) of the case study design situation. Solution knowledge in IDABES loosely corresponds to knowledge

about the design alternative (the solution) being developed in the case study design situation representation. Some of what is described as domain knowledge in IDABES fits part of that included in the design context (electrical network context) of the case study and the equivalents to the IDABES analysis algorithms also "fit" into the case study design situation representation at appropriate points in the design context. The qualitative commitments from the case study are not explicitly separately distinguished in IDABES. This brief comparison shows that constructive comparison is difficult as there is typically no useful correspondence between the components in different systems even if they are viewed at a fairly high level of (representation) abstraction.

Possibly the only realistic way for researchers to communicate their experience to one another in order to learn from each other's experiences at this level is to discuss and attempt to tackle the problems and shortcomings identified by each other's approaches. For instance, returning to the work of Doheny and Monaghan, they cite component configuration constraints as difficult to elicit " this knowledge is not well documented and exists in the minds of our design experts". This might provide an initial point of contact, a fruitful focus for discussion and experiment with the knowledge elicitation methods used for this thesis (for example).

Progress in disseminating and evolving understanding about engineering design and how it can be supported by knowledge based systems technology can also be approached by attempting to resolve different views about representing design at a more abstract level, for example to compare and test models of the design process. Smithers[1], for example, has expressed interest in exploring the extent to which the model of design as exploration which he has proposed (Smithers, 1990, 1990a) can account for the designing described in the case study of this thesis and the extent to which it is useful to view it in such a way. Dialogue at this level is more likely to lead to an increased understanding between researchers whose backgrounds and motivations may be very different and to result in fruitful new research programmes than is dialogue focused on comparison at the "implementation" level as described above.

The state of the art in supporting real design activity with knowledge based systems technology is at a far more immature stage of development than might be believed from a cursory inspection of the literature. Consequently there are very many ways in which the work presented in this thesis can be developed further; some suggestions about how some aspects of work presented might be extended are now given.

One of the least well understood areas in knowledge based systems research generally and in their application to design support particularly is that of knowledge elicitation. A "cook book" approach towards elicitation and representation of knowledge appropriate for designing which is

---

[1]Discussion with the author in June 1992.

suitable for application to all, or even an identifiable class, of design domains is still beyond the horizon. Empirical studies of designers, including the one presented here do nothing to suggest that such an approach is getting nearer nor do they support any notion that they are even likely to be found. However, more work of the kind reported here, suitably directed, will help to build up a body of experience with different design domains which may help to identify and refine suitable tools and formalisms to improve the success of, and possibly speed up and de-skill, the process of eliciting and representing design knowledge in a form suitable for knowledge based design support system implementation.

A system based on a competence model makes its components both explicit and open to inspection, thus, there is further scope for work which develops this approach because it holds promise for contributing to research on knowledge based systems in a number of areas. For example, the Competent Systems approach could be applied to the study of how knowledge based design support systems can assist with communication between designers or between designers and those who apply their designs.

Finally, focusing more specifically on the research reported here, it would be valuable to extend the work of exploring how support for design justification can be extended to see the extent (for example) to which a knowledge based system based on a competence model provides a medium for recording and conserving the rationale for designs (e.g. Boose, 1992). An associated line of further work would be to investigate further into how knowledge about designer's appreciative systems can be elicited and the extent to which it can be represented for effective use in generating, evaluating and justifying design decision.

REFERENCES

Adey 1993 — Adey,R.A., (ed.) *Artificial Intelligence in Design*, Progress in Engineering Series, Volume 12, Computational Mechanics Publications, Southampton, 1993.

Aikins 1983 — Aikins,J.S. "Prototypical knowledge for expert systems", *Artificial Intelligence*, **20**, 1983, pp.163-210.

Akin 1986 — Akin,O., *Psychology of Architectural Design*, Pion Limited, London, 1986.

Akin 1988 — Akin,O., "Expertise of the Architect" in Rychener 1988 op.cit., pp.173-196.

Alexander 1964 — Alexander,C., *Notes on the Synthesis of Form*, Harvard University Press, 1964.

Alexander 1988 — Alexander,C., "A City is not a Tree" in Thakara 1988 op.cit., pp.67-84.

Al Shawi 1990 — Al Shawi,S. & Quigley,G., "Experience of mediating Representation in an Industrial Application" in Addis,T. & Muir,R.M. (eds.), *Research and Development in Expert Systems VII*, Cambridge University Press,1990, pp.138-146.

Althaus 1989 — Althaus,K. & Backhouse,J., "An Expert System for the Modelling of Legal Norms" in Doukidis 1989 op.cit., pp.313-325.

Althusser 1972 — Althusser,L., *For Marx*, Pantheon, 1972.

Alvey 1982 — The Report of the Alvey Committee, *A Programme for Advanced Information Technology*, DTI, HM Stationery Office, 1982.

Asimow 1962 — Asimow,M., *Introduction to Design*, Prentice Hall, 1962.

Barrow 1985 — Barrow,H., "Proving Correctness in Digital Designs" in Bobrow,D.G. (ed.), *Qualitative Reasoning About Physical Systems*, MIT Press, Cambridge,Mass., 1985, pp.437-491.

Barthes 1972 — Barthes,R., *Critical Essays*, Northwestern University Press, 1972.

Barthes 1972a — Barthes,R., *Mythologies*,. Cape, 1972.

Bernstein 1985 — Bernstein,R.J., "From Hermeneutics to Praxis" in Hollinger 1985 op.cit., pp.272-296.

Bliss 1983 — Bliss,J., Monk,M., & Ogborn,J., *Qualitative Data Analysis for Educational Research,* Croom Helm, 1983.

Bodker 1991 — Bodker,S., *Through the Interface: A Human Activity Approach to User Interface Design*, Lawrence Erlbaum, 1991.

Boose 1992 — Boose,J.H., Bradshaw,J. & Sheema,D.B., "Knowledge-based Design Rationale Capture: Automating Engineering Trade Studies" in Green 1992, op.cit., pp.197-214.

Breuker 1987 — Breuker,J. & Wielinga,B.," Knowledge Acquisition as Modelling Expertise: the KADS Methodology", *Proc. 1st European Workshop on Knowledge Acquisition for Knowledge Based Systems*, Reading, England, 1987.

Broadbent 1979 — Broadbent,G., "The Development of Design Methods", *Design Methods and Theories,* **vol.13 no.1**,1979, pp.41-45 (reproduced in Cross 1984 op.cit.).

Brown 1981 — Brown,J.S., Burton,R.R. & deKleer,J., "Pedagogical, natural language and knowledge engineering techniques in SOPHIE I II and III" in Sleeman,D.H. (ed.), *Intelligent Tutoring Systems*, Academic Press, pp.227-282, 1981.

Brown 1984 — Brown,D.C. "Expert Systems for Design problem Solving using Design Refinement with Plan Selection and Redesign" *PhD Thesis*, Ohio State University, 1984.

Brown 1985 — Brown,D.C. & Chandrasekaran,B., "Plan Selection in Design Problem Solving" *Proc. of AISB 85*, (The Society for AI and the Simulation of Behaviour), 1985, pp.108-124.

Brown 1986 — Brown,D.C. & Breau,R., "Types of Constraints in Routine Design Problem-Solving" in Sriram 1986 op.cit., pp.383-390.

Brown 1986a — Brown,D.C. & Chandrasekaran,B., "Knowledge and Control for a Mechanical Design Expert System", *IEEE Expert*,, July 1986, pp.92-100.

Brown 1989 — Brown,D.C. & Chandrasekaran,B., *Design Problem*

Solving: Knowledge Structures and Control Strategies, Pitman, London, 1989.

Brown 1992
Brown,A.D., "Grounding Soft Systems Research", *Eur.J.Inf.Systs.*, **Vol.1, No.6**, 1992, pp.387-395.

Bundy 1979
Bundy,A., Byrd,L., Luger,G., Mellish,C., Milne,R. & Palmer,M., "Solving Mechanics Problems Using Meta-Level Inference", *Proc. IJCAI-79*, Tokyo, 1979, pp.1017-1027.

Chandrasekaran 1982
Chandrasekaran,B. & Mittal,S., "Deep Versus Compiled Knowledge Approaches to Diagnostic Problem Solving", *Proc. of AAAI-82*, 1982, pp.349-354.

Chandresekaran 1983
Chandrasekaran,B., "Towards a Taxomony of Problem Solving Types", *AI Magazine*, winter/spring 1983, pp.9-17.

Chandresekaran 1983a
Chandresekaran,B. & Mittal,S., "Conceptual Representation of Medical Knowledge for Diagnosis by Computer: MDX and Related Systems" in *Advances in Computers*, **Vol.22**, 1983, pp.217-293.

Chandrasekaran 1986
Chandrasekaran,B., "Generic Tasks in Knowledge-Based Reasoning: High-Level Building Blocks for Expert System Design", *IEEE Expert*, Fall 1986, pp.23-30.

Chandresekaran 1989
Chandresekaran,B., Josephson,J., Keuneke,A., & Herman,D., "Building Routine Planning Systems and Explaining their Behaviour", *International Journal of Man-Machine Studies*, **30** 1989,pp.377-398.

Chandrasekaran 1991
Chandrasekaran,B., "Models versus Rules, Deep versus Compiled, Content versus Form : Some distinctions in knowledge systems research", *IEEE Expert*,, April 1991, pp.75-79.

Clancey 1981
Clancey,W.J. & Leitsinger,R., "NEOMYCIN : reconfiguring a rule based expert system for application to teaching", *Proc. IJCAI-81*, 1981, pp.829-835.

Clancey 1983
Clancey W.J., "The epistemology of a rule based system : A framework for explanation", *Artificial Intelligence*, **20**, pp.215-251.

Clancey 1985 — Clancey,W.J., "Heuristic Classification",*Artificial Intelligence* ,**27**, pp.289-350.

Cohen 1987 — Cohen,R., May,J.H. & Pople,H.E., "An Intelligent Workstation for Electocentre Design", *I.E.E.E. Transactions on Systems, Man and Cybernetics*, **vol.SMC-17,no.2**, March/April 1987, pp.240-249.

Copplestone 1991 — Copplestone,T., *Twentieth Century World Architecture*, Brian Trodd Publishing House, London, 1991.

Coyne 1990 — Coyne,R., Rosenman,M.A., Radford,A.D., Balachandran,M., & Gero,J., *Knowledge-Based Design Systems*, Addison-Wesley, 1990.

Crinnon 1991 — Crinnon,J., *Evolutionary Systems Development*, Pitman, 1991.

Cross 1981 — Cross,N., Naughton,J. & Walker,D., "Design Method and Scientific Method", *Design Studies*, **vol.2 no.4**,1981, pp.195-201.

Cross 1984 — Cross,N. (ed.), *Developments in Design Methodology*, Wiley, 1984.

Cross 1989 — Cross,N., *Engineering Design Methods* ,Wiley, 1989.

Darke 1979 — Darke,J., "The Primary Generator and the Design Process", *Design Studies* **1 (1)**, 1979, pp.36-44.

Derrington 1987 — Derrington,P., "Management of the Design Process" in Sriram 1987 op.cit., pp.19-33.

DOE 1990 — Department of Energy, *Electricity Act 1989 Successor Company Licences in England and Wales Vol. 1 Public Electricity Supply Licence*, HMSO, 1990.

Doheny 1993 — Doheny,J.G. & Monaghan,P.F., "IDABES: An expert system for the preliminary stages of conceptual design of building energy systems" in Adey 1993 op.cit., pp.62-72, 1993 (first published in *Artificial Intelligence in Engineering*, Vol.2, No.2, 1987).

Doukidis 1989 — Doukidis,G., Land,F. & Millar,G. (eds.), *Knowledge Baseed Management Support Systems*, Ellis Horwood, 1989.

Dreyfus 1981 — Dreyfus,H., "From Microworlds to Knowledge Representation: AI at an Impasse" in Haugeland 1981 op.cit., pp.161-204.

Dreyfus 1986 — Dreyfus,H.L. & Dreyfus, S.E., *Mind Over Machine The Power of Human Intuition and Expertise in the Era of the Computer,* Macmillan, 1986 (page nos. referenced refer to the edition published by Free Press ,1988).

Dym 1991 — Dym,C. & Levitt,R.E., *Knowledge Based Systems in Engineering*, McGraw-Hill, 1991.

Ehn 1991 — Ehn,P., *Work-Oriented Design of Computer Artefacts,* Lawrence Erlbaum, 1991.

Electricity Association 1978 — "Engineering Recommendation P. 2/5 Security of Supply", Electricity Association Services, Ltd., London, 1978.

Engelmore 1988 — Engelmore,R. & Morgan,T. (eds.), *Blackboard Systems,* Addison Wesley, 1988.

Ericsson 1984 — Ericsson, K.A. & Simon,H.A., *Protocol Analysis Verbal Reports as Data*, M.I.T. Press, 1984.

Feltovitch 1984 — Feltovitch,P.J., Johnson,P.E., Moller,J.H. & Swanson,D.B., "LCS: The Role and Deveolpment of Medical Knowledge in Diagnostic Expertise" in Clancey,W.J. & Shortliffe,E.H. (eds.), *Medical Artificial Intelligence The First Decade*, Addison Wesley, 1984 pp.275-319.

Firth 1930 — Firth,J.R., *Speech,* Ernest Benn, 1930 (republished as Firth 1964).

Firth 1937 — Firth,J.R., *The Tongues of Men*, Watts &Co., 1937 (republished as Firth 1964).

Firth 1957 — Firth,J.R. *Papers in Lingusitics 1934-1957*, Oxford University Press, 1957.

Firth 1964 — Firth,J.R., *The Tongues of Men and Speech*, Oxford University Press, 1964.

Fischer 1989 — Fischer,G., McCall,R. & Morch,A., "Design Environments for Constructive and Argumentative Design", *Proc. of CHI'89* (Austin, Texas), ACM Press, 1989, pp.269-275.

Fischer 1989a — Fischer,G. & Nieper-Lemke,H., "HELGON: Extending the Retrieval by Reformulation Paradigm", *Proc. of CHI'89* (Austin, Texas), ACM Press, 1989, pp.357-362.

Fischer 1990 — Fischer ,G. & Girgensohn,A., "End-user Modifiability in Design Environments", *Proc. of CHI'90* (Seattle, Washington), ACM Press, 1990, pp.183-191.

Fischer 1990a — Fischer,G., Lemke,A. & Mastaglio,T., "Using Critics to Empower Users", *Proc. of CHI'90* (Seattle, Washington), ACM Press, 1990, pp.337-347.

Fischer 1991 — Fischer,G., Lemke,A. & Mastaglio,T., "Critics: an emerging approach to knowledge-based human-computer interaction", *Int.J. Man-Machine Studies*, **35**, 1991, pp.695-721.

Fischer 1991a — Fisher,G. & Nakakoji,K., "Empowering Designers with Integrated Design Environments" in Gero 1991 op.cit., pp.191-209.

Fischer 1991b — Fischer,G. & Nakakoji,K., "Making Design Objects Relevant to the Task at Hand", *Proc. 9th Nat. AAAI-91 Conference*, 1991, pp.67-73.

Fischer 1992 — Fischer,G. & Nakakoji,K., "Beyond the macho approach of artificial intelligence: empower human designers - do not replace them", *Knowledge-Based Systems*, **Vol 5 No 1**, 1992, pp.15-30.

Florman 1976 — Florman,S.C., *The Existential Pleasures of Engineering*, Barrie and Jenkins, London, 1976.

Flurscheim 1977 — Flurscheim,C., *Engineering Design Interfaces A Management Philosophy*, Design Council, 1977.

Foucault 1972 — Foucault, M., *The Archaeology of Knowledge*, Pantheon, 1972.

Fox 1992 — Fox,M.S., Finger,S., Gardner,E., Navin Chandra,D., Safier,S.A. & Shaw,M., "Design Fusion: An Architecture for Concurrent Design" in Green 1992, op.cit., pp.157-195.

Funes 1993 — del Valle Funes,M., *The Determination of Relevance Through Metaphor in Knowledge Acquisition*, PhD. Thesis, Brunel University, 1993.

Frayman 1987 — Frayman,F. & Mittal,S., "COSSACK: A Constraint-Based Expert System for Configuration Tasks" in Sriram 1987 op.cit., pp.143-166.

Gadamer 1975 — Gadamer,H.G., *Truth and Method*, Sheed and Ward, 1975.

Gadamer 1981 — Gadamer,H.G., *Reason in the Age of Science*, M.I.T. Press, 1981.

Galal 1993 — Galal,G.H. & McDonnell,J.T., "Using Systemic Grammar Networks for Knowledge Elicitation: a Methodological Perspective", *Proc. 13th International Conference on Artificial Intelligence, Expert Systems and Natural Language, Avignon'93*, pp.75-84.

Gero 1990 — Gero,J., "Design Prototypes : A Knowledge Representation Schema for Design", *AI Magazine* **11(4)**, Winter 1990, pp.26-36.

Gero 1991 — Gero,J. (ed.), *Artificial Intelligence in Design '91*, Butterworth-Heinemann, 1991.

Gero 1992 — Gero,J. (ed.), *Artificial Intelligence in Design '92*, Kluwer Academic Publishers, Holland, 1992.

Gilb 1988 — Gilb,T., *Principles of Software Engineering Management*, Addison Wesley, 1988.

Gordon 1961 — Gordon,W.J.J., *Synetics, the Development of Creative Capacity* , Harper, New York, 1961.

Graham 1990 — Graham,D., *A Case Study in Computer Fault Diagnosis and Repair*, PhD. Thesis, Brunel University, 1990.

Green 1992 — Green,M., *Knowledge Aided Design*, Academic Press, 1992.

Gregory 1966 — Gregory,S. (ed.), *The Design Method*, Butterworths, 1966.

Gregory 1978 — Gregory,M. & Carroll,S., *Language and Situation Language Varieties and their Social Contexts*, Routledge & Kegan Paul, 1978.

Gutierrez 1987 — Gutierrez,O., "Some Aspects of Information Analysis Using a Repertory Grid Technique" in Galliers,R.(ed.), *Information Analysis*, Addison Wesley,1987, pp.347-362.

Halliday 1978 — Halliday,M., *Language as a Social Semiotic*, Edward Arnold, 1978.

Hammer 1990 — Hammer,M., "Re-engineering Works - Don't Automate, Obliterate", *Harvard Business Review*, July/Aug., 1990, pp.104-112.

Hammond 1989 — Hammond,K., *Case-Based Planning : Viewing Planning as a Memory Task* , Academic Press, 1989.

Harri-Augstein 1991 Harri-Augstein,S. & Thomas,L., *Learning Conversations*, Routledge, 1991.

Hasling 1984 Hasling,D.W., Clancey,W. & Rennels,G., "Strategic explanations for a diagnostic consultation system", *Int.J.Man-Machine Studies*, **20**, 1984, pp.3-19.

Haugeland 1981 Haugeland,J. (ed.), *Mind Design*, MIT Press, 1981.

Hayes-Roth 1983 Hayes-Roth, F., Waterman,D.A. & Lenat,D.B., *Building Expert Systems*, Addison-Wesley, 1983.

Hesse 1974 Hesse,J., *The Structure of Scientific Inference*, Macmillan, 1974.

Hillier 1972 Hillier,W., Musgrove,J., & O'Sullivan,P., "Knowledge and Design" in Mitchell,W.J. (ed.), *Environmental Design and Practice*, University of California, 1972.

HMSO 1990 see DOE 1990.

Hogley 1986 Hogley,J.R. & Korncoff,A.R., Artificial Intelligence in Engineering: A Revolutionary Change, *Proc. Ist International Conference on Applications of A.I. in Engineering* (Southampton), **vol.2**, 1986, pp.1155-1160.

Hollinger 1985 Hollinger,R., *Hermeneutics and Praxis*, University of Notre Dame Press, 1985.

Johnson 1984 Johnson,L. & Hartley,R.T., "Conversation Theory and Cognitive Coherence Theories", *Cybernetics and Systems Research* **2**, pp.647-650.

Johnson 1985 Johnson,N.E., "Varieties of Representation in Eliciting and Representing Knowledge for IKBS", *Int.J.System Research and Info.Science*, **vol. 1**,1985, pp.69-90.

Johnson 1986 Johnson,L. & Keravnou,E.T., "Competent Expert Systems: A Framework for More Adequate Explanations", *Second Alvey Workshop on Explanations*, Surrey University, March 1986.

Johnson 1986a Johnson,L. & Keravnou,E.T., "Analysing, Representing and Interpreting Expert Strategic Knowledge" in R.D.Trappl (ed.), *Cybernetics and Systems*, Reidel Publishing Co, 1986, pp.743-750.

Johnson 1987 Johnson,L., & Johnson,N., "A Knowledge Elicitation Method for Expert Systems Design", *Int.J.System*

Research and Info.Science, **vol. 2**, 1987, pp.153-166.

Johnson 1990 — Johnson,N.E., Tomlinson,C. & Johnson,L., "Second Generation Expert Systems and Knowledge Elicitation", *Int.J.System Research and Info.Science*, **vol. 4**, 1987, pp.87-99.

Johnson 1991 — Johnson,L., "Informatics and Practical Reasoning", *Int.J.Sys.Research and Info.Science*,**Vol 4**, 1991, pp 207-216.

Jones 1970 — Jones,J.C., *Design Methods Seeds of Human Futures*, Wiley, 1970.

Jones 1991 — Jones,J.C., *Designing Designing*, Architecture Design and Technology Press, London, 1991.

Kassalty 1987 — Kassalty,A. & Brown,D.C., Explanation for Routine Design Problem Solving in Sriram 1987 op.cit., pp.225-239.

Kearney 1986 — Kearney,R., *Modern Movements in European Philosophy*, Manchester University Press, 1986.

Kelly 1955 — Kelly,G., *The Psychology of Personal Constructs*, Norton, 1955.

Keravnou 1986 — Keravnou,E.T. & Johnson,L., *Competent Expert Systems A Case Study in Fault Diagnosis*, Kogan Page, 1986.

Keravnou 1989 — Keravnou,E.T. & Washbrook,J., "What is a Deep Expert System? An Analysis of the Architectural Requirements of Second Generation Expert Systems", *The Knowledge Engineering Review*, **no.4:3**, 1989, pp.205-233.

Kidd 1987 — Kidd,A.L. (ed.), *Knowledge Acquisition for Expert Systems A Practical Handbook*, Plenum Press, 1987.

de Kleer 1986 — de Kleer,J., "An Assumption-Based TMS", *Artificial Intelligence*, **28**, 1986, pp.127-162.

Koestler 1964 — Koestler,A.,*The Act of Creation*, Hutchinson, 1964.

Koestler 1967 — Koestler,A.,*The Ghost in the Machine*, Hutchinson, 1987.

Kolb 1984 — Kolb,D.A., *Experiential Learning Experience as the Source of Learning and Development*, Prentice Hall, 1984.

Kolb 1990 — Kolb,D., *Postmodern Sophistications Philosophy*,

Architecture and Tradition, University of Chicago Press, 1990.

Kunz 1970 — Kunz,W. & Rittel,H., *Issues as Element of Information Systems*, Working Paper 131, Centre for Planning and Development Research, University of California, Berkley, 1970.

Kwauk 1988 — Kwauk,R. & Brown,D.C., "Generating and Applying Failure Recovery Suggestions in Hierarchical Design Systems" in Gero,J. (ed.)*Artificial Intelligence in Engineering: Diagnosis and Learning* , Elsevier 1988, pp.29-50.

Lacan 1977 — Lacan,J., *The Four Fundamental Concepts of Psychoanalysis*, Penguin, 1977.

LaFrance 1990 — LaFrance,M., "The Special Structure of Expertise" in McGraw 1990 op.cit., pp.55-70.

Latombe 1976 — Latombe,J-C., "Artificial Intelligence in Computer-Aided Design: the TROPIC System", *AI Technical Note 125*, Stanford Research Instutute, Menlo Park, California, 1976.

Lawson 1990 — Lawson,B., *How Designers Think The Design Process Demystified* (2nd Ed.), Butterworth, 1991.

Lenat 1990 — Lenat,D.B., Guha,R.V., Pittman,K., Pratt,D. & Shepherd,M., "Cyc: Towards Programs with Common Sense", *Communications of the A.C.M.*, **vol 33,no.8**, August 1990, pp.30-49.

Levi-Strauss 1963 — Levi-Strauss,C., *Structural Anthropology*, Basic Books, 1963.

Lewis 1991 — Lewis, P.J., "The decision making basis for information systems: the contribution of Vicker's concept of appreciation to a soft systems perspective" *Eur.J.Inf.Systs.*, Vol.1,No.1, 1991, pp.33-43.

LeCorbusier 1927 — Le Corbusier, *Towards a New Architecture* (trans. Etchells,F.), Rodker, London, 1927.

Logan 1989 — Logan,B., "Conceptualizing Design Knowledge", *Design Studies*, **Vol.10,No.3**, 1989, pp.188-195.

Logan 1991 — Logan,B., Millington,K. & Smithers,T., "Being Economical with the Truth: assumption-based context management in the Edinburgh Designer System" in Gero 91 op.cit., pp.423-446.

Logan 1992 — Logan,B., Corne,D.W. & Smithers,T., "Enduring Support : on defeasible reasoning in design support systems" in Gero 1992 op.cit., pp.433-454.

Logan 1992a — Logan,B.,Corne,D. & Smithers,T., "Using Reason Maintenance Systems to Support Ill Structured Problem Solving", *Proc. ECAI'92*, 1992, pp.282-286.

Logan 1992b — Logan,B. & Smithers,T., "Creativity and Design as Exploration" in Gero,J.S. & Maher,M-L. (eds.), *Modelling Creativity and Knowledge-Based Creative Design*, Lawrence Erlbaum,New Jersey, 1992, pp.149-188.

Malinowski 1935 — Malinowski,B. *Coral Gardens and their Magic*, Allen & Unwin, 1935.

March 1976 — March,L.J., "The Logic of Design and the Question of Value" in March,L.J. (ed.), *The Architecture of Form*, Cambridge University Press, 1976 (reproduced in Cross 1984 op.cit.).

Marcus 1989 — Marcus,S. & McDermott,J., "SALT: A Knowledge Acquisition Language for Propose-and-Revise Systems", *Artificial Intelligence* **39**, 1989, pp.1-37.

Mayall 1979 — Mayall,W.H., *Principles in Design*, Design Council, 1979.

McCall 1989 — McCall,R., "MICROPOLIS: A hypertext system for design", *Design Studies*, **Vol 10 No 4**, 1989, pp.228-238.

McCall 1991 — McCall,R., "PHI: a conceptual foundation for design hypermedia", *Design Studies*, **Vol 12 No 1**, 1991, pp.30-41.

Meyer 1988 — Meyer,B., *Object-Oriented Software Construction*, Prentice Hall, 1988.

McDonnell 1986 — McDonnell,J.T., *Knowledge Acquisition and Implementation of a Prototype for an Expert System to Design Biochemical Reactors*, MSc Thesis, Department of Computer Science, University College, London, 1976 (submitted under name J.T.Sinclair).

McDonnell 1991 — McDonnell,J. & Johnson,L., "Modelling Competence in Engineering Design: Implications for Expert System Architectures",*Int.J.System Research and Info.Science*, **Vol. 4**, 1991, pp.217-228.

McDonnell 1991a McDonnell,J.T., "Modelling Competence in Knowledge based Systems for Power Systems Applications", *Proc. Third Symposium on Expert Systems Applications to Power Systems*, Tokyo-Kobe, April 1-5, 1991, pp.526-529.

McDonnell 1993 McDonnell,J.T. & Johnson,L., "What are Adequate Explanation Facilities? Explanation and Justification in Expert Systems",*Int.J.Sys.Research and Info.Science*,**Vol 6**,1993 , pp. 37-47.

McGraw 1990 McGraw, K.L. & Westphal,C.R., *Readings in Knowledge Acquisition Current Trends and Practices*, Ellis Horwood, 1990.

Merleau-Ponty 1962 Merleau-Ponty,M., *The Phenomenology of Perception*, Routledge, 1962.

Mittal 1984 Mittal,S. , Chandrasekaran,B. & Sticklen,J., "PATREC: A Knowledge-Directed Data Base for a Diagnostic Expert System" *IEEE Computer*, **17(9)**, 1984, pp.51-58.

Monaghan 1986 Monaghan,P.F. & Doheny,J.G., "Knowledge Representation in the Conceptual Design Process for Building Energy Systems" in Sriram 1986 op.cit., pp.1187-1192.

Mostow 1985 Mostow,J., "Towards Better Models of the Design Process" *A.I.Magazine*, **vol.6**, 1985, pp.44-56.

Murdoch 1985 Murdoch,S., "Intelligent Databases for Expert Systems", *Int.J. Systems Research and Information Science* **vol1 no.3**, 1985, pp.145-162.

Murdoch 1990 Murdoch,S. & Johnson,L., *Intelligent Data Handling*, Chapman & Hall, 1990.

Newell 1967 Newell,A., Shaw,J.C. & Simon,H.A., "The Process of Creative Thinking" in Gruber,H., Terrell,G. & Wertheimer,M. (eds.), *Contemporary Approaches to Creative Thinking*, Atherton Press, New York, 1967, pp.63-119.

Newell 1982 Newell,A. "The Knowledge Level", *Artificial Intelligence*, 18, 1982, pp87-127.

Newton 1988 Newton,S. & Logan,B.S., "Causation and its effect:the blackguard in CAD's clothing", *Design Studies*, **Vol.9,No.4**, pp.196-201,1988.

Ogborn 1984 Ogborn,J.M. & Johnson,L., "Conversation Theory",

Kybernetes, **Vol. 13**, 1984, pp.7-16.

Osborn 1957 — Osborn,A.F., *Applied Imagination - Principles and Procedures of Creative Thinking*, Scribner, New York 1957.

Pahl 1988 — Pahl,G. & Beitz,W., *Engineering Design: A Systematic Approach*, The Design Council, London, 1988.

Pask 1975 — Pask,G., *Conversation, Cognition and Learning: A Cybernetic Theory and Methodology*, Elsevier, 1975.

Pask 1976 — Pask,G., *Conversation Theory Applications in Education and Epistemology*, Elsevier, 1976.

Petroski 1985 — Petroski,H., *To Engineer is Human : The Role of Failure in Successful Design*, Macmillan, 1985.

Petroski 1989 — Petroski,H., "Failure as a unifying theme in design", *Design Studies*, **Vol.10, No.4**,1989, pp.214-218.

Pidgeon 1991 — Pidgeon,N.F., Turner,B. & Blockley,D.I., "The use of Grounded Theory for conceptual analysis in knowledge elicitation", *Int.J. Man-Machine Studies*, **35**, 1991, pp.151-173.

Popper 1963 — Popper,K.R., *Conjectures and Refutations*, Routledge and Kegan Paul, 1963.

Popper 1968 — Popper,K.R., *The Logic of Scientific Discovery* Hutchinson, London, 1968.

Popplestone 1980 — Popplestone,R.J., Amber,A.P. & Bellos,I., "An Interpreter for a Language for Describing Assemblies", *Artificial Intelligence*, **14(1)**, 1980, pp.79-107.

Potter 1980 — Potter,N., What is a Designer? (2nd.ed.), Hyphen Press, Reading, England, 1980.

Pye 1964 — Pye,D., *The Nature of Design* ,Studio Vista, London, 1964.

RIBA 1965 — *Architectural Practice and Management Handbook*, RIBA Publications, London, 1965.

Ricoeur 1981 — Ricoeur,P., *Hermeneutics and the Human Sciences*, Cambridge University Press, 1981.

Rittel 1973 — Rittel,H. & Webber,M.M., "Dilemmas in a General Theory of Planning" *Policy Sciences* **4**, 1973 pp.155-169 (partly reproduced in Cross 1984 op.cit.).

Robins 1980 — Robins, R.H., *General Linguistics An Introductory Survey* (3rd ed.), Longman, 1980.

Rogers 1983 — Rogers,G.F.C., *The Nature of Engineering,* Macmillan, 1983.

Rowe 1987 — Rowe,P., *Design Thinking,* MIT Press, 1987.

Rumbaugh 1991 — Rumbaugh,J., Blaha,M., Premerlani,W., Eddy,W. & Lorensen,W., *Object-Oriented Modelling and Design,* Prentice Hall, 1991.

Rychener 1988 — Rychener,M.D. (ed.), *Expert Systems for Engineering Design,* Academic Press, 1988.

Ryle 1949 — Ryle,G., *The Concept of Mind*, Penguin Books, 1949.

Ryle 1954 — Ryle,G., *Dilemmas*, Cambridge University Press, 1954.

Saussure 1916 — Saussure,F. de, *Cours de Linguistique Générale*, published in English as *Course in General Linguistics*, The Philosophical Library, 1959.

Schön 1983 — Schön,D., *The Reflective Practitioner How Professionals Think in Action* ,Basic Books, 1983.

Schön 1985 — Schön,D., *The Design Studio An Exploration of its Traditions and Potential,* RIBA Publications, London 1985.

Schön 1987 — Schön,D., *Educating the Reflective Practitioner*, Jossey-Bass, California, 1987.

Schön 1992 — Schön,D.A., "Designing as a Reflective Conversation with the Materials of a Design Situation", *Knowledge Based Systems*, **Vol.5(1)** March 1992, pp.3-14.

Schön 1992a — Schön,D. & Wiggins,G., "Kinds of seeing and their functions in designing", Design Studies, Vol.13(2), April 1992, pp.135-156.

Sembugamoorthy 1986 — Sembugamoorthy,V. & Chandrasekaran,B., "Functional Representation of Devices and Compilation of Diagnostic Problem Solving Systems" in Kolodner,J.L. & Riesbeck,C.K. (eds.), *Experience, Memory and Reasoning*, Lawrence Erlbaum, NJ, 1986, pp.47-73.

Shaw 1981 — Shaw,M.L.G. (ed.), *Recent Advances in Personal*

Construct Technology, Academic Press, 1981.

Shaw 1987 — Shaw,M.L.G. & Gaines,B.R., "An Interactive Knowledge Elicitation Technique using Personal Construct Technology" in Kidd 1987 op.cit., pp.109-136.

Simon 1969 — Simon,H., *The Sciences of the Artificial* , MIT Press, 1969.

Smithers 1986 — Smithers,T. & Potton,S., "The Alvey "Design to Product" Large Scale Demonstrator", *Proc. Int. Conf. on Computer-Aided Production Engineering*, Edinburgh, April 1986, pp.311-317.

Smithers 1989 — Smithers,T., "Intelligent Control in AI-Based Design Support Systems", *DAI Research Paper No.423*, Department of Artificial Intelligence, University of Edinburgh, 1989.

Smithers 1990 — Smithers,T., Conkie,A., Doheny,J., Logan,B., Millington,K. & Tang,M.X., "Design as Intelligent Behaviour: an AI in Design Research Programme", *Artificial Intelligence in Engineering*, **Vol.5, No.2**, 1990, pp.78-109.

Smithers 1990a — Smithers,T. & Troxell,W., "Design Is Intelligent Behaviour, But What's the Formalism?", *Artificial Intelligence in Engineering Design and Manufacturing*, **Vol.4(2)**, 1990, pp.89-98.

Smithers 1992 — Smithers,T.,Tang,M.X., Tomes,N., Buck,P., Clarke,B., Lloyd,G., Poulter,K., Floyd,C. & Hodgkin,E., "Development of a Knowledge-based Design Support System", *Knowledge-Based Systems*, **Vol.5,No.1**, 1992, pp.31-40.

Southwick 1991 — Southwick,R.W., "Explaining Reasoning: An Overview of Explanation in Knowledge-Based Systems", *The Knowledge Engineering Review*, **No.6:1**, 1991, pp.1-19.

Sriram 1986 — Sriram,D. & Adey,R. (eds.), *Applications of A.I. in Engineering Problems*, Springer-Verlag, 1986.

Sriram 1986a — Sriram,D. & Maher,M.L., "The Representation and Use of Constraints in Structural Design" in Sriram 1986 op.cit., pp.355-368.

Sriram 1987 — Sriram,D. & Adey,R. (eds.), *Knowledge Based Expert Systems in Engineering: Planning and Design*, Computational Mechanics Publications, 1987.

Stamper 1987 — Stamper,R.K., Althaus,K. & Backhouse,J.P., "Survey of the LEGOL/NORMA Projects", *London School of Economics Papers in Infomatics*, LSE, 1987.

Steels 1988 — Steels,L., *Components of Expertise AI Memo No 88-16*, Artificial Intelligence Laboratory, Vrije University, Brussels,1988.

Steiner 1978 — Steiner,G., *Heidegger*, Fontana Press, 1978.

Stewart 1981 — Stewart,V. & Stewart,A., *Business Applications of Repertory Grid*, McGraw Hill, 1981.

Strauss 1990 — Strauss,A. & Corbin,J., *Basic Qualitative Research: Grounded Theory Procedures and Techniques*, Sage Publications, 1990.

Stylianou 1991 — Stylianou,A.K., *Object Orientation for Information Representation and Management*, PhD. Thesis, Brunel University, 1991.

Suchman 1987 — Suchman,L.A., *Plans and Situated Actions : The Problem of Human-Computer Communication*, Cambridge University Press,1987.

Swartout 1989 — Swartout,W., *Producing Explanations of Expert Systems 1*, Morgan Kaufmann (video), 1989.

Thakara 1988 — Thakara,J. (ed.), *Design After Modernism*, Thames and Hudson, 1988

Thomas 1985 — Thomas,L. & Harri-Augstein,E.S., *Self-Organised Learning, Foundations of a Conversational Science for Psychology*, Routledge & Kegan Paul, 1985.

Tomlinson 1993 — Tomlinson,C.M., *The Analysis and Synthesis of Distributed Knowledge Using the Johnson Methodology*, PhD. Thesis, Brunel University, 1993.

Tong 1992 — Tong,C. & Sriram,D., *Artificial Intelligence in Engineering Design*, Academic Press, 1992.

Tunnicliffe 1990 — Tunnicliffe, A.J., *Knowledge Elicitation in Design: A Case Study of Page Layout Design* , PhD. Thesis University of Loughborough, 1990.

Tunnicliffe 1992 — Tunnicliffe, A.J. & Scrivener, S.A.R., *Knowledge Elicitation in Design*, Design Studies, Vol.12 No.,2, April 1991, pp.73-80.

Vickers 1965 — Vickers,G., *The Art of Judgement A Study of Policy*

Making, Chapman Hall, 1965.

Vickers 1968 Vickers,G., *Value Systems and Social Process*, Tavistock Publications, London, 1968.

Visser 1991 Visser,W., "The Cognitive Psychology Viewpoint on Design: Examples from Empirical Studies" in Gero 1991 op.cit.,pp.505-524.

Visser 1992 Visser,W., "Designers' activities examined at three levels: organization, strategies and problem-solving processes", *Knowledge Based Systems*, Vol.5, No. 1, March 1992, pp.92-104.

Watts 1966 Watts,R.D., "The Elements of Design" in Gregory 1966 op.cit., pp.85-95.

Weber 1978 Weber,M., *Ecomony and Society Vol.2,* Roth.G. & Wittich,C. (eds.), University of California Press, 1978.

Windsor 1989 Windsor,C., *A Vision of Britain A Personal View of Architecture, the Prince of Wales* , Doubleday, 1989.

Winograd 1986 Winograd,T. & Flores,F., *Understanding Computers and Cognition : A New Foundation for Design*, Ablex Publishing, 1986.

Wittgenstein 1953 Wittgenstein L., *Philosophical Investigations*, Blackwells, Oxford, 1953.

APPENDIX 1

Notes from Interview 4

Note : This record is in the form of notes taken from the tape recording of the interview because the interview consisted of a repertory grid exercise. This is best described through non-verbatim transcription since there is (on the tape) constant reference to proposals laid out on the desk e.g. "these two are alike and that one is different because ..".

The interview begins with an explanation given by the interviewer of the repertory grid exercise procedure.

4.1

Seven proposals have been made available. The interviewer decided to use six for the exercise - to make the exercise easier - six being a suitably rounded number for the grid exercise. The proposal excluded was the West Central London one which is much larger in every respect than the others.

The six proposals (names disguised) used were :

(A) Swedenill Diversion (referred to in chapter 7)
(B) Breedpace Lane Reinforcement (referred to in chapter 7)
(C) Thire
(D) Silnoting Substation
(E) Shifer Street
(F) Aklodo Common.

4.2

(Ideally for the repertory grid technique it might be thought that more than six elements should be used. However an over-riding consideration in this approach is that the elements should be "items of personal experience" and thus it is essential that the elements be relevant to the person from whom constructs are being elicited. The proposals which were used were all the results of planning work which the subject had personally carried out or had directed and represented problem situations with which he was very familiar.)

The "personal construing" aspect of the exercise was emphasised to the subject before commencing so that he should not pre-filter anything that occurred to him on a mistaken assumption about what sort of constructs were appropriate.

Interviewer: "It's very much a personal thing, so say what you think. So if it's something that doesn't seem to be related to them (the proposals) that's ok, but that makes them seem the same, like you were in a different office when you were doing two of the , something like that".

Subject: "it's the old objective that this is subjective then". 4.3

In preparation, to ensure reasonably even coverage, the interviewer had grouped the six proposals into different combinations of three. The

intention was to cycle round these six in different combinations until nothing more usefully (i.e. different) seemed to be emerging. This was explained to the subject.

Notes were made on cards as the exercise progressed to keep track of what emerged and through which combination of elements. 4.4

A,B and C

(1) straight replacement/obsolete components (A,C)
no obsolescence (B)

(2) no appreciable load growth (A,C)
pure load growth (B)

(3) potential problems remain after work (A,C)
may well have to go back (A,C)
problem completely solved (B)
immediate problem solved (B)
stop gap solution (B) (qualified by :
to be swept up with something else (B) eventually after discussion below:

Subject: "If you have to go back at another time (A,C) what you've done (the current work) will be relevant to what you then have to consider"

(Comment on exercise: Subject: "I'd rather stop at that stage because I think otherwise (pause) that's really the salient things, those are the salient features that hit you first thing. I mean the more and more you think about it, I mean you can keep on coming up with other reasons." (later) "It may well be that something that's said later on, about these in relation to another will cover the same point.")

(Observation: subject is anxious that he will "run out" of material.) 4.5

D,E and F

(Comment on exercise: Interviewer: "I think it's actually quite helpful to actually see them in front of you isn't it?" Subject: "Well, yeah, it refreshes my memory, that's more to the point..."Those two go together.".)

(Observation: (on last phrase above) Intonation shows subject quickly warming the exercise and its challenge - contrasts with his concerns expressed over this exercise at end of interview 3.) 4.6

(4) pure obsolescence (D,F)
pure load related (E)

(5) stop gap (short term) (E,F)
long term solution (D)

The term "stop gap" may be used to indicate a short term solution because some new development which will affect it is known to be coming along. But in terms of obsolescence (D) is a long term solution (it solves this aspect long term) i.e. long term solution is not opposite to stop gap but a stop gap solution is equivalent to a short term one. This distinction was arrived at later through discussion (see below). 4.7

(Observation: The above shows the quality of what can be drawn out in such an exercise.)

(Observation: The subject has "jumped ahead" in describing (5) since points (4) and (5) represent different pairings, he quickly "gets the hang" of the exercise with a little conversational prompting.)

(Comment on exercise: the subject admits at this point to wanting to "save something" for the later comparisons in the exercise.) 4.8

(6) one option solution (effectively) (E)

In the proposal two schemes were outlined but the subject recalls the proposal as being unusual in their being only one option - on the basis that the alternative proposed was not a "realistic" one. It was put in to show (rehearse the argument) that it was not feasible. In that respect it was just a document formatting choice that resulted in the argument being put as a separate option, it could equally well have been dismissed as a possible realistic option by discussion elsewhere in the text of the report. This is why the subject thinks of this proposal as having only one option.

4.9

B, D and F

Subject:	"Basically you've got the same groupings and the same reasons as for Shifer Street (E), it's a dead ringer."
Interviewer:	"so we're just substituting." - i.e. (B) for (E) in comparison above - same comparisons. - This turns out to be true for the notion of obsolescence vs. load growth but then things get interesting :
Subject:	You could group those two (B and F) as a second grouping.In as much as these are stop gap solutions - that's a long term (D). (Pause).
Interviewer:	So actually Breedpace Lane (B) is moved now, because at the beginning we were thinking that that was, um, (pause) um, we were saying that Breedpace Lane (B) wasn't actually stop gap then weren't we?
Subject:	Um

Interviewer: Because with the Swedenill Diversion and Thire (A and C) you had that (sic) potential problems could arise still - you were leaving old equipment there, whereas with Breedpace Lane (B), right, it's stop gap in the sense that, um, you're..

Subject: We're waiting for the West City Centre (scheme).

Interviewer: ..waiting for West City Centre, but, but, er, it's, um, you're not leaving any possibility there of faulty equipment or anything like that.

Subject: No, um.

Interviewer: So we've got a sort of different thing there.

Subject: You've got a very slight different, different biassing. Um, so ok, um, I see what you're doing now I see how you're exploring this.

There then follows a discussion to differentiate between the apparently two different purposes for which the term "stop gap" is being used. "Stop gap" is a qualifier e.g. on load growth or obsolescence. 4.10

(7) consumer driven (F)

There will be a need to go back there (F) because the work is arising out of a particular consumer's requirements. The other two (B,D) - Subject: "there won't be anything going on there in terms of obsolescence for 15 to 20 years". 4.11

(Comment on exercise: subject's interest grows rapidly, particularly after the "stop gap" revelation above - see subject's comment in context above he says "I see what you're doing now I see how you're exploring this." and follows with these further comments shortly afterwards: On finding similarities - "I literally am playing all sorts of games with this." and " (pause) ..you can play these in all directions when you want to" (clearly enjoying the exploration).) 4.12

A discussion about the proposal for Breedpace Lane (B) ensues to clarify the role of the development plans for the West City Centre generally (development plan document referenced) in determining what is proposed for Breedpace Lane (B). The interviewer suggests that the scheme for Breedpace Lane mentions (development plan document) to show that the proposals "fit in" with the general development plan. The subject does not approve this but is able to suggest that "does not run contrary to" is a better way of capturing the intention.

The interviewer asks the subject to give an illustration of a way in which the proposal at Breedpace Lane (B) could have been a poor proposal on the

grounds of running contrary to the general development plan. The subject thinks about this for a while and then uses the security of supply standard to give an explanation. 4.13

(half a page is omitted here from the original notes for confidentiality reasons)

He then comes back to the example (Aberdeen Place (B)) and explains that it was a two transformer set up and therefore required particular circuit inter-connections to be able to satisfy the second circuit outage (a double circuit outage to the transformers - would require the load to those two transformers to be transferred away) restoration standards. It is by meeting the security standard that the proposal doesn't contravene the plans for development of the central London area. He says that the economics of the scheme are affected by the need to meet the security standard. The standard over-rides financial considerations therefore. 4.14

This prompts comments on constraints.

Subject: "We've made it fit, we've made it fit in what is not the best way of doing it, in terms of economic network design, it's not the best. It's not something I would have liked to have done if we'd had a free hand,

(rest of explanation from original notes omitted for confidentiality reasons)

So -, if you like, um, outside constraints of the security standard, has introduced, has affected the economics, of that particular scheme."

Interviewer: "So, I mean it always affects, it's always underlying things, but in that case you can see it quite specifically, that you've had to accommodate it."

Subject: "We've had to accommodate, and of-course, the other thing, space requirements, at Breedpace Lane (B) we hadn't got space for any more than two transformers. And we also physically could only get single bus-bar equipment in. That's not such a massive constraint, because I could have got round, if I could have got four transformers in or three transformers in, I could have got round the single bus-bar problem. There's ways of getting round that. But it's really the fact that we haven't got any more space to put in any more capacity than what we've actually put in. So there's a whole series of constraints, perhaps not very well spelt out. Um, I can't remember, I'd actually have to go back to the report to see if we actually spelt it out well in there or not. But they're constraints which are, which I'm fully aware of, if you like from background and experience." 4.15

(Comment on exercise: this shows how deeper knowledge can be elicited almost as asides to the repertory grid mainstream activity - the subject can see this happening. For example following the remarks above on constraints he says "That's probably put the cat among the pigeons.") 4.16

C, A and E

(8) obsolescence problem (C,A)
load related problem (E)

(9) single option solution (E)

(10) stop gap - waiting for West City Centre (E)

(11) potential problems remain (C, A)
shouldn't be any problem for 20 years (E)

In the case of (C) the age of the cables feeding the substation and the auto-transformers, for (A) the fact that some old cable remains on the route. Whereas (E) is a 1960s substation.

Subject further notes that he can find nothing in common between (C) and (E) or (A) and (E). 4.17

Subject: "..which is strange actually, that's the first time, (pause) that's the first time I haven't been able to ring the changes."

4.18

Interviewer: "Let's see if we can look at why that might be then, why they're so different." The subject then reflects for some time eventually coming up with what follows below.

One proposal is an absolute immediate problem (E) - a pressing load growth problem. The other two (C and A) have no immediate loading problems associated. This leads on to consideration of how proposals arise - some have to be rushed through - (E) and (A) (and (B) to a lesser extent) had to be done "very, very quickly". Discussion reveals that this is a matter of months and it is also possible that the draft proposal stage is omitted. The subject arrives at the conclusion that a rushed job would be one that did not appear in the rolling five-year plans (budgeting cycle - giving rise to the capital programme). A rushed job would not appear in the allowance for capital expenditure. Interviewer then asks about (A), (B) and (C) w.r.t. the capital programme. (A) and(B) were not included but (C) was.

4.19

(Comment on exercise - discussion has therefore led to consideration of A, B and C - a shift which the subject notices. (The interviewer was aware of the shift as it happened.)

4.20

This gives us:

A, B and C

(12) some proposals are rushed through which means they do not appear in the capital expenditure plan 4.21

F, E and B

(13) load growth problems (E, B)
immediate problems (E, B)
pure obsolescence problem (F)

(14) deal with immediate problem, and
fall within the West City Centre remit (E, B)
geographically remote (F) but see (15)

A question from the interviewer about what geographically remote means leads to a definition of the area included as (affected by) the West City Centre plans.

Subject: (describes boundary by reference to prominent geographical features - rivers, parks, etc. actual words in original notes omitted here)

4.22

(15) stop gap solutions (all three F, E, and B)
- for different reasons : load growth (E and B) consumer driven (F)
- these then all linked by subject to idea that they are all immediate problems. 4.23

C, E and D

(Comment made that this combination has already been covered by deviations earlier.)

D, F and A

(16) obsolete due to age (D, F)
obsolete due to "manufacturing" (A) unexpectedly obsolete 4.24

(Observation: The subject clearly had an immediate inclination to make this split but took some time (silently) to think about why it had been made. The subject was clearly surprised at what he found out by the reflection. It was quite clear that the subject would not have thought of this distinction spontaneously.)

Note: This information is an example of how this technique complemented the drawing of systemic grammar networks since this refinement of "obsolescence" helps to develop the network (cf. figure 6.5 in chapter 6). 4.25

(17) "expediency" / cheapest solution (A, F)
tidy solution / better engineering solution (D)

The reasons for arriving at the solution for (F) were consumer-driven. In the case of (A) cheapest "no provision for tomorrow" solution. In the case of (D) the old switchgear and building is got rid of although not the transformers. To make a tidy solution all the switchgear was cleared (some could have been kept) but that would have been "untidy" leaving 30 year old gear. In explaining this further (on further probing from interviewer) the subject refers to their having been "advanced expenditure" to produce the tidy solution. 4.26

(Comment on exercise: at this point we have:

Interviewer: "We might want to probe what a tidy solution was (sic) at some point."

Subject: "Yes that's what I said, you know, that's an interesting point, this is an intriguing game to actually put two together like that."

Interviewer: " Yes, it does bring things out, I think, which.."

Subject: "It brings things out that I must confess I wouldn't have thought of."

At this point the subject is very keen to proceed and takes charge of moving on to the next trio.) 4.27

(E, C, and A)

This is the last group, nothing new emerges, we get obsolescence for (C, A) vs. purely load related (E). Also we get immediate problems (E, A) and a problem "we have lived with for quite a while" (C). This leads to a volunteered explanation of switchgear with no rating - switchgear not rated to BS116, 1937 - "it's that obsolete". If there is an accident there would be problems over Health and Safety regulations. Switchgear needs to be operated, if it has no rating there are operating constraints on it - one aspect of this is that of reducing the fault level to a minimum - at least while someone is in the switch room, sometimes no one is allowed in the switch room. 4.28

Note: This clarification of the notion of obsolete switchgear is another example of information that can be incorporated into a refinement of the systemic grammar network.

(END OF TAPE 4)

A little over one hour was spend on the exercise reported above. About 8 combinations of groups of three proposals were formally considered, but as can be seen from the above the conversation led to other comparisons, equally fruitful, through matters arising. It was felt that even an enjoyable exercise (and the subject clearly did enjoy this one) has its useful length remembering that the subject has to "think hard" while he is doing it. A follow up date (a week later) was arranged for the grid focusing step of the procedure (notes of this next meeting are given as appendix 2).

4.29

END OF NOTES OF INTERVIEW 4

APPENDIX 2

Notes from Interview 5

These notes and transcribed extracts are from the fifth interview which continues and concludes the repertory grid exercise begun a week earlier in interview 4 (given as appendix 1). In the intervening time the interviewer has produced a raw grid showing the constructs elicited in the first session and the elements (which of the six proposals) they arose from initially. The objective in this interview is to explore the extent to which the elicited constructs apply to each element. In exploring this it is expected that the constructs will be refined further. 5.1

The interviewer shows the subject a (large) grid with each element (each of the six proposals) heading a column and with the constructs forming poles at each end of each row. The cells contain an indication of which pole of the construct applies for each element (where known). The grid has been partially focused before being drawn up for this meeting.

The interviewer explains about the refinement of the constructs and the intention to "fill in" the empty cells where possible.

The cells are not considered individually one by one, instead the interviewer has examined the material from the previous session and guides the conversation in the direction of the most promising "grey areas" or the biggest "holes" in the grid. For example she starts with: 5.2

Interviewer: " Something that seemed very interesting that came up was this idea of identifying something as being an expediency sort of solution, in other words you did the cheapest thing at the time."

The subject finds it difficult initially to plunge straight into this so the interviewer elaborates by recalling aloud the (three) specific instances (elements) where this appeared to apply. This helps the subject to get back into the topic.

In the case of proposal F the (planner's (representing the company's)) objective was to get a potential fault ("a disastrous failure"), due to old equipment, eliminated as soon as possible (fault would have resulted in loss of supply to part of the city's underground transportation system). Whereas the customer was seeking a delay "in case the problem was overtaken by events" i.e. customer wanted to delay expenditure because of possible future developments which would allow the problem to be solved differently. 5.3

In the case of proposal A, expediency/ cheapest solution is explained by looking at "the rest of the equipment tacked onto it". The view was taken that this equipment had another 20 years of life. The view was taken that "the cables will probably stand up to it as well " i.e. last 20 years too. It was concluded that expediency was used here to mean the question "What is the

justification for spending (additional) money now?" had been answered by a decision not to spend money now for investment in something that may not give benefit for 20 years (new cables). 5.4

This discussion prompted the subject to suggest that expediency was perhaps not a good description. The interviewer asks him to suggest something better. He thinks about this for a while. The interviewer suggests that perhaps the two cases are two dissimilar (i.e. the situations labelled expediency in proposal A and F). The interviewer suggests consideration of the third case (proposal C).

In the case of C it was:

Subject: "very much a tidy solution, you got rid of a lot of old equipment." It was tidy because, although they had "gone half way" i.e. replaced some equipment and left some.

Subject: "we had left our options open, it's actually possible to develop the system" i.e. if load growth appeared suddenly, it could be dealt with. Also the replacement transformers were second hand, they have a limited life, if the old cables (feeding them) failed the transformers could be replaced.

Subject: "All the options are open to move forward if something else develops at Thire (C), that's why, if you like, I call it tidy." 5.5

Interviewer: " So really, if somebody came to you and showed you that as a solution, you'd recognize it as being an elegant one."

Subject: " Yeah, it's got the attraction of elegance because it's capable of, (slight pause) it's capable of further development, capable of further expansion. Whereas anything, I know I used the word last week, stop gap solutions, they literally are filling, filling a particular hole at that moment in time, and you know it may be possible to expand them, but it's not so easily visible, and you're looking, and I know we've used the term tidy solution, um, and that's an expression that we've used, it may be that the attraction of that is that, from a personal point of view, it's a good engineering solution, whereas a stop-gap solution, because it's, because it doesn't appear to lend itself to expansion, variation, doesn't hold the same sort of engineering appeal." 5.6

The interviewer shifts the conversation to allow consideration of some of the other proposals (elements) which did not originally give rise to the current construct to see how these other elements can be classified (to fill in the grid cells) and to stimulate the discussion in this area further:

Interviewer: "Let's look at Shifer Street (E) from that same perspective". There was really only one option on that scheme.

A discussion of "cheapness" ensues. The interviewer suggests that a cheap solution is one where the cost is the over-riding consideration in the decision. The subject feels however that cost is always of importance in the sense that "you are always looking for the cheapest solution anyway in overall terms, but there can, in some instances, you can give, if you like, more weighting to engineering factors. So although you can argue that you go for the cheapest solution, you can sometimes weight it out (sic) with engineering factors, to, um, to try and offset the cheapest, if, you like, but I think I would also argue that a good engineering scheme should also be the cheapest anyway." The subject suggests that the term cheapest is perhaps not appropriate. He tries to think of something better, but is struggling so the interviewer suggests they consider another proposal from this perspective (B). 5.7

In the case of B it is "still possible to move forward", it is still possible to "develop those elements (meaning bits of equipment etc.) in a different way".

Subject: "if you had done, using the definition of exped ... and cheapest solution scenario, there would probably be no way forward on that, but there's potential on the basis of the land we've got out there and the way we've set the Breedpace Lane “B” Substation thing up, is to use elements of the Breedpace Lane “B” Substation, to overcome future problems on Breedpace Lane “A” (proposal B), albeit that they might be 10, 15, 20 years ahead. And I think this is probably, in a round about way, I think we're actually getting to really what I mean, we're actually getting a truer definition of cheapest solution. I mean we keep on saying here tidy, it's a tidy solution, and what I keep on saying to you is that there's, it appears to be a good engineering solution."

(There was then a short interruption while the subject dealt with some pressing request from a third party.) 5.8

Subject recaps by saying that with a good engineering solution, a tidy solution, he can "see his way back at some time in the future to develop the system further" whereas in some cases the solution that has been adopted "does not lend itself to development, you almost have to go back to square one". There is no longer term benefit, he feels this is closely related to the idea of stop gap. He feels that expediency and stop gap should apply to the same situations.

There is a short discussion about immediate problems and how these do not

necessarily lead to short term solutions. 5.9

Comment on exercise (by the subject) at this point : Subject "..this is my first reaction, the way you've set this out (referring to the grid produced) you are actually beginning to get some sort of linking between some of these..". 5.10

The topic changes to a discussion of the term "advanced expenditure". The subject explains that it means that "we recognize that in 5 or 10 years time, 5, 10, 20 years time, a piece of equipment will become obsolete and we would have to spend money to replace it." So to achieve a tidy or optimum solution (subject's terms) money will be spent now (rather than in 20 years time). It is not often done because it is difficult to justify the expenditure. The subject speculates that this will become increasingly difficult with the new methods of calculating cost benefit. 5.11

The discussion moves to the consideration of the idea of long term and short term solutions and the link with the notion of awaiting future developments (several proposals would be affected by broader plans for the western part of the city centre). The proposals (elements) giving rise initially to this construct were quickly reviewed, B and E were awaiting city centre developments. Whereas F was a short term consumer driven solution and Thire (C) was seen to be a tidy solution which dealt with an immediate obsolescence problem because it would be possible to "do something" should load develop, in that sense it could be seen as a long term solution. (Its long termness springs from its ability to be adapted or extended later if needed, "it's capable of development".) 5.12

In the case of A (filling in the gaps in the grid) the subject found it difficult to categorise this in terms of long term and short term. It was definitely not a solution awaiting future developments. Future developments would be independent. It's a solution to "get you over an immediate problem". 5.13

In the case of D, it would tend to be a long terms solution - the subject gives his reasons for this thus: "Again you haven't got any problems with load growth, you've still got some old transformers there, um, (slight pause) you've dealt with the immediate problems, you've still got some old transformers there, and you've still got some old stuff behind it. And it may well be that you'd go in and do something completely different, um, again it doesn't really fall into either category (i.e. long term or short term) rather like the Swedenill Diversion (A) because, um, you know you're going to take three steps back if you do get other problems in that area... .. you may have to start completely from scratch and ignore what you've got there." 5.14

There is then a shortish aside while the subject explains about new "portable" switch rooms which are supplied in portacabins which can be moved elsewhere once they are no longer required at a particular site "you know it's the caravan principle". This arrangement had been used for C and D. So although the solutions are thought to be long term, if they turn out not to be so "there's room for manoeuvre". It's not long term from the elegant, tidy perspective but from the room for manoeuvre one. 5.15

There is brief discussion now of how this is beginning to give some more focusing to the grid, more links at least. 5.16

The interview then continues the interviewer still trying to fill in some of the biggest gaps in the grid. The interviewer is interested in the idea of solutions where there were potential problems remaining vs. problems which were completely solved. First, however she asks a specific question about use of the term "obsolete" as applied to different items. The subject explains that in the case of switchgear usually it means it hasn't got a rating to BS116 1937, or there can be something specifically wrong with the design of the equipment. He stresses the importance of differentiating among these two explaining that if there is no rating to BS116 and something goes wrong in its operation there is likely to be a prosecution if someone is hurt. Whereas equipment with a design fault may possibly be compensated for by introducing an "operating regime". For transformers, obsolete is usually a reference to an obsolete design, so for its size and rating it is either high loss or requires high maintenance, the tap change mechanism is old and worn and may be causing problems, spare parts are difficult to obtain. So the term obsolete applied to a transformer is far more generalised than when applied to switch gear where it is something specific. For cables, it may be due to a manufacturing problem, but usually it can't be said to be obsolete since cable is still in operation that is 90 years old. Unless there is an identifiable problem one does not talk about cable being obsolete. Proposals C and D were dealing with obsolete switchgear. At C potential problems remain due to the age of the cables and the auto-transformers. In the case of F the switchgear was unrated. The subject explains that "unrated" means there is no record of it having being tested to BS117, an assigned rating is sometimes given and for many years switchgear was operated on the basis of an assigned rating, now things are tightened up and this practice is avoided. The interviewer begins to ask why a decision was made to do something about Thire (C) at a particular time (as it had presumably been lived with for a long time with unrated switchgear installed) but there is an interruption to the interview (which lasts about 10 minutes). 5.17

On resumption, the interviewer suggests that they try to fill in some of the gaps in the grid.

At this point the conversation becomes less discursive, each new element and construct being dealt with fairly swiftly without significant new points coming up.

5.18

For example:
Shifer Street (E) had no obsolescence in the immediate future, also D. At F the cables are old (1920s) so there is a potential problem but "you need to be careful because there's so much consumer effect on that one". No cable problem at E. E is a short term solution but is looking towards the west city centre work. The whole problem is solved but it's only an immediate problem.

They proceed to fill in gaps straightforwardly now by just deciding which

pole of the constructs applies to the remaining elements. Some of the constructs do not really apply to some elements, these are also noted. When most of the gaps have been filled in on the semi-focused grid the interviewer decides to stop the exercise having obtained a reasonable amount of data to permit further refinement of the grid constructs and further focusing. 5.19

There then follows a brief conversation about what the subject of the next meetings shall be as follows:

Firstly the interviewer asks about the proposal mentioned in a previous meeting which the subject said had 6 or more options. It was a job at Branse, un-published. Privatisation had caused the original plans to be revised. (For one thing the discount rate had been changed from 8% to 17.5%.) It is agreed that the subject will talk the interviewer through the various options at the next meeting. Incidentally it was noted that the external factors related to privatization would probably have resulted in the recommendations of different options for each of the proposals which had been studied during the meetings thus far. 5.20 The interviewer expresses an interest in seeing how some of the 6 options are to be eliminated to form the basis for the published version of the proposal for Branse. (Note: This forms much of the basis for discussion at interview 6.) 5.21

The interviewer also asks how the large proposal for the West City Centre should be tackled. The subject suggests that the proposal should be studied by the interviewer before the meeting where he will answer questions on it. (Note: This is discussed at interview 7.)

The interviewer arranges to borrow copies of all the proposals which have been discussed in the meetings so far.

5.22

END OF INTERVIEW 5.

APPENDIX 3 : Appearance of the repertory grid during focusing.

		(A)	(C)	(D)	(F)	(E)	(B)	
(1)	no appreciable load growth	←	←	←	←	→	→	pure load growth/ load related problem
(2)	(purely) obsolete components	← manuf. fault	← age of s/w gr- untestable	← age of s/w gr- unrateable	← age of s/w gr- unrated	→	→	no obsolescence
(3)	"tomorrow's" loading problems	←	←	⇢	n/a	→	→	"today's" (urgent) loading problem
(4)	potential problems remain (due to condition of cable)	← cable	← & auto t/fs & supplying cable	← infeeds use old t/fs	← old cables remain see foot of col.	→ no further possibilities	→	problem completely solved
(5)	"special" security constraints & affected by plans for city centre	→	→	→	→	←	←	standard security constraints, or geographically remote
(6)	short term solution (*awaiting future developments)	←	→ flexible - moveable equip.	⇢	← (*)	← (*)	← (*)	long term solution
(7)	stop gap, for immediate problems	n/a	→ but obsol. immed. prob.	n/a	←	←	← immed. prob. but not stop gap	known (for long time) problem
(8)	rushed job, (not in capital programme, no draft proposal)	←	→	→	→	←	←	in capital programme
(9)	cheapest solution - expediency	← no justification for replacing s/w gr.	n/a tidy soln. but no advance expenditure	→ advanced expenditure	← consumer driven (single consumer bears cost)	⇠ a "one option only" solution	⇢	tidy solution, flexible -potential for further development

APPENDIX 4

Information about the Implementation Platform - KAPPA-PC

The components of the knowledge based design support system described in chapter 8 were implemented in KAPPA-PC (version 2.0) running under Microsoft Windows Version 3.0. KAPPA-PC is a product of Intellicorp, Inc[1].

The environment provided by KAPPA-PC supports object-oriented software development. Applications are build by specifying object hierarchies which can be viewed and developed either by using a graphical development interface or by writing code in a traditional way using a programming window. Part of the environment consists of an application development language, KAL, which provides conventional programming constructs and list processing facilities. These are included in the approximately two hundred and fifty functions of the KAL language.

The developers' interface conforms in style to personal computer based windows software. Facilities are provided to develop and view object hierarchies, to program directly using the KAL language, to edit object classes and instances, and to build up user defined functions, rules and goals. The interface offers browsing, tracing and debugging facilities to improve the speed and quality of application development.

Windows-style graphical user interfaces can be built for applications by creating instances of session windows incorporating instances of buttons, icons, dialogue boxes, pull-down menus, text display panels, etc. which are pre-defined as generic objects as part of KAPPA-PC's graphical interface development environment.

A variety of external interfaces are provided to other common PC-based software such as spreadsheets, database management systems and word processors.

KAPPA-PC is not restricted in suitability to "expert systems" or "knowledge based systems" applications. Rules and goals can be coded to produce applications which are equivalent to those which can be produced on more restricted "expert system shell" types of platform. However the main value of KAPPA-PC is as a much more flexible software engineering environment offering object-based programming constructs, comprehensive debugging support and high level building blocks for object-based user interface development.

---

[1]Intellicorp, Inc., 1975 El Camino Real West, Mountain view, CA 94040-2216, U.S.A.

For the work presented in chapter 8 most emphasis was on exploiting the facilities to define, edit and develop object hierarchies, to specify slots and methods in the object classes, to make use of the inheritance mechanism and to achieve system functionality primarily by message passing between objects. A rudimentary developer's interface was developed using the graphical interface building facilities. No use was made of KAPPA rules or goals. Facilities for producing text files were used to provide documentation of what has been implemented, apart from this no external interfaces to other software were exploited.

The recommended support for running KAPPA-PC is 2Mb RAM, about 3Mb of storage space, EGA or VGA graphics, a mouse, MS-DOS Version 3.0 or higher and Microsoft Windows Version 3.0 or higher.

APPENDIX 5

Implementation KAL Source Code

Only that source code relevant to the implementation described in chapter 8 is listed here, all code associated with the developer's interface is omitted for clarity.

```
/*************************************************************
ALL CLASSES REFERENCED IN CHAPTER 8
ARE SHOWN BELOW
*************************************************************/

/*****************************************
**** CLASS: QualComm
*****************************************/

MakeClass( QualComm, Root );

/************** METHOD: DisplaySelf  **************/

MakeMethod( QualComm, DisplaySelf, [role ],
   {
    ShowWindow( Session4 );
    Global:tClass = Self;
    DisplayText( Transcript1, role, " qualities are ",
        FormatValue( "\n " ), Self:Description, FormatValue( "\n ( " ),
       Self:Rationale, ") " );
    IncreaseTraceIndentation(  );
    IndentTrace(  );
    DisplayText( Transcript4, "Exploring ", role, " qualities:  ",
        Self:Description, FormatValue( "\n " ) );
   } );

/************** METHOD: ShowParent  **************/

MakeMethod( QualComm, ShowParent, [],
    DisplayText( Transcript2, "The more general quality is : ", FormatValue(
"\n" ),
        GetValue( GetParent( Self ), Description ), " - ",
```

```
        FormatValue( "\n   " ), GetValue( GetParent( Self ), Rationale ),
        FormatValue( "\n" ) ) );

/*************** METHOD: ShowChildren   ***************/

MakeMethod( QualComm, ShowChildren, [],
   {
    ResetValue( Global:tList );
    GetSubClassList( Self, Global:tList );
    If ( LengthList( Global:tList ) == 0 )
     Then {
          DisplayText( Transcript2, "There are no more specific qualities. ",
            FormatValue( "\n" ) );
       }
     Else {
          DisplayText( Transcript2, "More specific qualities : ",
            FormatValue( "\n" ) );
          EnumList( Global:tList, item, DisplayText( Transcript2,
                                item:Description,
                                 " - ", FormatValue( "\n   " ),
                                 item:Rationale, FormatValue( "\n" ) ) );
       };
    ResetValue( Global:tList );
   } );

/*************** METHOD: ShowSiblings   ***************/

MakeMethod( QualComm, ShowSiblings, [],
   {
    ResetValue( Global:tList );
    GetSubClassList( GetParent( Self ), Global:tList );
    If ( LengthList( Global:tList ) < 2 )
     Then {
          DisplayText( Transcript2, "There are no alternative contributing
qualities to same more general quality. ",
            FormatValue( "\n" ) );
       }
     Else {
          DisplayText( Transcript2, "Alternative contributing qualities to same
more general quality : ",
            FormatValue( "\n" ) );
          EnumList( Global:tList, item, If ( item #= Self )
                                Then NULL
                                 Else DisplayText( Transcript2,
                                      item:Description,
                                       " - ", FormatValue( "\n   " ),
                                      item:Rationale,
                                       FormatValue( "\n" ) ) );
       };
    ResetValue( Global:tList );
   } );
```

```
MakeSlot( QualComm:Rationale );
QualComm:Rationale = "Enable system to be managed efficiently within
regulations";
SetSlotOption( QualComm:Rationale, IF_NEEDED, NULL );
SetSlotOption( QualComm:Rationale, WHEN_ACCESS, NULL );
SetSlotOption( QualComm:Rationale, BEFORE_CHANGE, NULL );
SetSlotOption( QualComm:Rationale, AFTER_CHANGE, NULL );

MakeSlot( QualComm:Description );
QualComm:Description = "Develop an efficient system";
SetSlotOption( QualComm:Description, IF_NEEDED, NULL );
SetSlotOption( QualComm:Description, WHEN_ACCESS, NULL );
SetSlotOption( QualComm:Description, BEFORE_CHANGE, NULL );
SetSlotOption( QualComm:Description, AFTER_CHANGE, NULL );

MakeSlot( QualComm:ArgumentSummary );
SetSlotOption( QualComm:ArgumentSummary, MULTIPLE );
SetValue( QualComm:ArgumentSummary, "The electrical network should
deliver power efficiently.", "It must meet the licence conditions." );
SetSlotOption( QualComm:ArgumentSummary, IF_NEEDED, NULL );
SetSlotOption( QualComm:ArgumentSummary, WHEN_ACCESS, NULL );
SetSlotOption( QualComm:ArgumentSummary, BEFORE_CHANGE,
NULL );
SetSlotOption( QualComm:ArgumentSummary, AFTER_CHANGE, NULL
);

    /*************************************
     **** CLASS: ReduceLosses
     *************************************/

MakeClass( ReduceLosses, QualComm );
ReduceLosses:Rationale = "This is a licence condition";
ReduceLosses:Description = "Reduce System Losses.";
SetValue( ReduceLosses:ArgumentSummary, "System energy losses
should be considered when designing.", It, is, a, requirement, of, the,
licence, conditions, to, do, this. );

    /*************************************
     **** CLASS: SupplCloseLoad
     *************************************/

MakeClass( SupplCloseLoad, ReduceLosses );
SupplCloseLoad:Rationale = "Avoids voltage regulation problems.";
SupplCloseLoad:Description = "Place main substations close to load
centres.";
SetValue( SupplCloseLoad:ArgumentSummary, "Sources of supply (main
sub-stations) should be placed close to their loads.", "This avoids voltage
regulation problems and reduces energy losses." );
```

```
/*************************************
 **** CLASS: ReduceComplexity
 *************************************/

MakeClass( ReduceComplexity, QualComm );
ReduceComplexity:Rationale = "Makes network simpler to operate.";
ReduceComplexity:Description = "Reduce system complexity.";
SetValue( ReduceComplexity:ArgumentSummary, "Re-design or changes
to the network should reduce its complexity.", "This makes system
operation simpler." );

/*************************************
 **** CLASS: ReduceEquip
 *************************************/

MakeClass( ReduceEquip, ReduceComplexity );
ReduceEquip:Rationale = "Reliability increased through less equipment to
fail.";
ReduceEquip:Description = "Reduce amount of equipment.";
SetValue( ReduceEquip:ArgumentSummary, "Re-design of part of the
network should exploit opportunity to reduce the amount of equipment
installed.", "This increases system reliability." );

/*************************************
 **** CLASS: ElimIntermedTf
 *************************************/

MakeClass( ElimIntermedTf, ReduceEquip );
ElimIntermedTf:Description = "Eliminate intermediate transformation
points.";
SetValue( ElimIntermedTf:ArgumentSummary, "Opportunities to remove
intermediate stages of transformation should be taken.", "This simplifies
the electrical network.", "It increases reliability and reduces costs." );

/*************************************
 **** CLASS: ReplWithLess
 *************************************/

MakeClass( ReplWithLess, ReduceEquip );
ReplWithLess:Rationale = "Plant cost (per MVA) is reduced by using larger
capacity equipment.";
ReplWithLess:Description = "Replace cables or transformers with fewer of
higher capacity.";
SetValue( ReplWithLess:ArgumentSummary, "Opportunities to replace
equipment by larger units should be taken.", "Costs per unit of enery are
reduced in this way." );
```

```
/*************************************
 **** CLASS: InstalRadialNets
 *************************************/

MakeClass( InstalRadialNets, ReduceComplexity );
InstalRadialNets:Rationale = "Reduces time to reconfigure after fault. Less
skilled staff required to operate.";
InstalRadialNets:Description = "Install radial networks.";
SetValue( InstalRadialNets:ArgumentSummary, "Electrical networks
should be made radial when possible.", "They are easier to operate
particularly after a fault.", "They can be handled by less highly trained
staff." );

/*************************************
 **** CLASS: MaxFlexibility
 *************************************/

MakeClass( MaxFlexibility, QualComm );
MaxFlexibility:Rationale = "Defers reinforcement at high voltage.";
MaxFlexibility:Description = "Maximise flexibility.";
SetValue( MaxFlexibility:ArgumentSummary, "The electrical network
should be designed to maximise flexibility in the way it is supplied.", "This
defers reinforcement of the higher voltage network." );

/*************************************
 **** CLASS: SpreadLoadGrth
 *************************************/

MakeClass( SpreadLoadGrth, MaxFlexibility );
SpreadLoadGrth:Description = "Spread the potential for load growth.";
SetValue( SpreadLoadGrth:ArgumentSummary, "The electrical network
should be designed to spread load growth potential.", "This defers re-
designing of the higher voltage network." );

/*************************************
 **** CLASS: AllowLoadTfPot
 *************************************/

MakeClass( AllowLoadTfPot, MaxFlexibility );
AllowLoadTfPot:Description = "Allow for potential to transfer load.";
SetValue( AllowLoadTfPot:ArgumentSummary, "The network should be
designed to allow for load transfer potential.", "This defers reinforcement
at higher voltage." );

/*************************************
 **** CLASS: BalanceBSPLd
 *************************************/

MakeClass( BalanceBSPLd, MaxFlexibility );
BalanceBSPLd:Description = "Balance loading on supergrid supply
points.";
```

```
SetValue( BalanceBSPLd:ArgumentSummary, "Changes to the network
should keep the loading on bulk supply points balanced.", "This defers the
need for reinforcement at higher voltages." );


    /*************************************
     **** CLASS: DesAlt
     *************************************/

MakeClass( DesAlt, Root );

/*************** METHOD: SupportingQs  **************/

MakeMethod( DesAlt, SupportingQs, [],
   {
    SendMessage( Self:QualsSupporting, DisplaySelf, supporting );
   } );

/*************** METHOD: CounteringQs  **************/

MakeMethod( DesAlt, CounteringQs, [],
   {
    SendMessage( Self:QualsCountering, DisplaySelf, countering );
   } );

/*************** METHOD: RelevantQs  **************/

MakeMethod( DesAlt, RelevantQs, [],
   {
    SendMessage( Self:QualsRelevant, DisplaySelf, "other relevant" );
   } );

/*************** METHOD: MakeNewInstance  **************/

MakeMethod( DesAlt, MakeNewInstance, [],
   {
    Global:NewestInstance = ( Self # ( CountInstances( Self )
                                + 1 ) );
    MakeInstance( Global:NewestInstance, Self );
    AppendToList( Global:Instlist, Global:NewestInstance );
    IndentTrace( );
    DisplayText( Transcript4, GetValue( Global, NewestInstance ),
       FormatValue( "\n" ) );
   } );

/*************** METHOD: DispNetCntxAbstractSuggestion ***/

MakeMethod( DesAlt, DispNetCntxAbstractSuggestion, [],
   {
    ClearTranscriptImage( Transcript3 );
```

```
    EnumList( Self:NetCntxAbstractSuggestion, item,
       DisplayText( Transcript3, item, FormatValue( "\n" ) ) );
    IndentTrace( );
    DisplayText( Transcript4, "Suggested focus:", FormatValue( "\n" ) );
    EnumList( Self:NetCntxAbstractSuggestion, item,
      {
       IndentTrace( );
       DisplayText( Transcript4, item, FormatValue( "\n" ) );
      } );
   } );

/************** METHOD: TriggerNetCntxAbstract  **************/

MakeMethod( DesAlt, TriggerNetCntxAbstract, [],
  {
   ShowWindow( GetNthElem( GetValue( Self, NetCntxAbstractTrigger ),
            2 ) );
   IncreaseTraceIndentation( );
   IndentTrace( );
   DisplayText( Transcript4, "Following up suggestion. ",
      FormatValue( "\n" ) );
  } );

/************** METHOD: GetNetCntxConcrete  **************/

MakeMethod( DesAlt, GetNetCntxConcrete, [],
  {
   none;
  } );

/************** METHOD: DispNetCntxConcreteSuggestion  ***/

MakeMethod( DesAlt, DispNetCntxConcreteSuggestion, [],
  {
   DisplayText( Transcript3, FormatValue( "\n" ),
Self:NetCntxConcreteSuggestion,
      Self:NetCntxConcrete );
   IndentTrace( );
   DisplayText( Transcript4, Self:NetCntxConcreteSuggestion,
Self:NetCntxConcrete,
      FormatValue( "\n" ) );
  } );

/************** METHOD: IdentifyToTrace  **************/

MakeMethod( DesAlt, IdentifyToTrace, [],
  {
   IndentTrace( );
   DisplayText( Transcript4, Self:InstanceIdentifier,
      FormatValue( "\n" ) );
  } );
```

```
MakeSlot( DesAlt:CapitalCost );
SetSlotOption( DesAlt:CapitalCost, INHERIT, FALSE );
SetSlotOption( DesAlt:CapitalCost, VALUE_TYPE, NUMBER );
SetSlotOption( DesAlt:CapitalCost, MINIMUM_VALUE, 0 );
SetSlotOption( DesAlt:CapitalCost, MAXIMUM_VALUE, 10000000 );
SetSlotOption( DesAlt:CapitalCost, IF_NEEDED, NULL );
SetSlotOption( DesAlt:CapitalCost, WHEN_ACCESS, NULL );
SetSlotOption( DesAlt:CapitalCost, BEFORE_CHANGE, NULL );
SetSlotOption( DesAlt:CapitalCost, AFTER_CHANGE, NULL );

MakeSlot( DesAlt:StartDate );
SetSlotOption( DesAlt:StartDate, PROMPT, "When will this
recommendation become effective?" );
SetSlotOption( DesAlt:StartDate, IF_NEEDED, NULL );
SetSlotOption( DesAlt:StartDate, WHEN_ACCESS, NULL );
SetSlotOption( DesAlt:StartDate, BEFORE_CHANGE, NULL );
SetSlotOption( DesAlt:StartDate, AFTER_CHANGE, NULL );

MakeSlot( DesAlt:Duration );
SetSlotOption( DesAlt:Duration, ALLOWABLE_VALUES, less, than, 5,
years, 5, to, 10, years, 10, to, 15, years, up, to, 20, years, at, least, 20, years,
until, another, scheme, takes, over );
DesAlt:Duration = "at least 20 years";
SetSlotOption( DesAlt:Duration, PROMPT, "How long will this solution be
useful?" );
SetSlotOption( DesAlt:Duration, IF_NEEDED, NULL );
SetSlotOption( DesAlt:Duration, WHEN_ACCESS, NULL );
SetSlotOption( DesAlt:Duration, BEFORE_CHANGE, NULL );
SetSlotOption( DesAlt:Duration, AFTER_CHANGE, NULL );

MakeSlot( DesAlt:Status );
SetSlotOption( DesAlt:Status, ALLOWABLE_VALUES, recomended,
supporting, ruled, out, NULL );
DesAlt:Status = supporting;
SetSlotOption( DesAlt:Status, PROMPT, "Is this alternative recommended,
supporting a recommendation or ruled out? - enter recommended,
supporting or ruled out" );
SetSlotOption( DesAlt:Status, IF_NEEDED, NULL );
SetSlotOption( DesAlt:Status, WHEN_ACCESS, NULL );
SetSlotOption( DesAlt:Status, BEFORE_CHANGE, NULL );
SetSlotOption( DesAlt:Status, AFTER_CHANGE, NULL );

MakeSlot( DesAlt:QualsSupporting );
SetSlotOption( DesAlt:QualsSupporting, IF_NEEDED, NULL );
SetSlotOption( DesAlt:QualsSupporting, WHEN_ACCESS, NULL );
SetSlotOption( DesAlt:QualsSupporting, BEFORE_CHANGE, NULL );
SetSlotOption( DesAlt:QualsSupporting, AFTER_CHANGE, NULL );

MakeSlot( DesAlt:QualsCountering );
SetSlotOption( DesAlt:QualsCountering, IF_NEEDED, NULL );
```

```
SetSlotOption( DesAlt:QualsCountering, WHEN_ACCESS, NULL );
SetSlotOption( DesAlt:QualsCountering, BEFORE_CHANGE, NULL );
SetSlotOption( DesAlt:QualsCountering, AFTER_CHANGE, NULL );

MakeSlot( DesAlt:QualsRelevant );
SetSlotOption( DesAlt:QualsRelevant, IF_NEEDED, NULL );
SetSlotOption( DesAlt:QualsRelevant, WHEN_ACCESS, NULL );
SetSlotOption( DesAlt:QualsRelevant, BEFORE_CHANGE, NULL );
SetSlotOption( DesAlt:QualsRelevant, AFTER_CHANGE, NULL );

MakeSlot( DesAlt:LongName );
DesAlt:LongName = "Design Alternative";
SetSlotOption( DesAlt:LongName, IF_NEEDED, NULL );
SetSlotOption( DesAlt:LongName, WHEN_ACCESS, NULL );
SetSlotOption( DesAlt:LongName, BEFORE_CHANGE, NULL );
SetSlotOption( DesAlt:LongName, AFTER_CHANGE, NULL );

MakeSlot( DesAlt:NetCntxAbstractTrigger );
SetSlotOption( DesAlt:NetCntxAbstractTrigger, MULTIPLE );
SetValue( DesAlt:NetCntxAbstractTrigger, NetCntx, Session7 );
SetSlotOption( DesAlt:NetCntxAbstractTrigger, IF_NEEDED, NULL );
SetSlotOption( DesAlt:NetCntxAbstractTrigger, WHEN_ACCESS, NULL );
SetSlotOption( DesAlt:NetCntxAbstractTrigger, BEFORE_CHANGE, NULL
);
SetSlotOption( DesAlt:NetCntxAbstractTrigger, AFTER_CHANGE, NULL
);

MakeSlot( DesAlt:NetCntxAbstractSuggestion );
SetSlotOption( DesAlt:NetCntxAbstractSuggestion, MULTIPLE );
SetValue( DesAlt:NetCntxAbstractSuggestion, "*There is no suggested
(abstract) focus." );
SetSlotOption( DesAlt:NetCntxAbstractSuggestion, IF_NEEDED, NULL );
SetSlotOption( DesAlt:NetCntxAbstractSuggestion, WHEN_ACCESS,
NULL );
SetSlotOption( DesAlt:NetCntxAbstractSuggestion, BEFORE_CHANGE,
NULL );
SetSlotOption( DesAlt:NetCntxAbstractSuggestion, AFTER_CHANGE,
NULL );

MakeSlot( DesAlt:NetCntxConcrete );
SetSlotOption( DesAlt:NetCntxConcrete, IF_NEEDED, GetNetCntxConcrete
);
SetSlotOption( DesAlt:NetCntxConcrete, WHEN_ACCESS, NULL );
SetSlotOption( DesAlt:NetCntxConcrete, BEFORE_CHANGE, NULL );
SetSlotOption( DesAlt:NetCntxConcrete, AFTER_CHANGE, NULL );

MakeSlot( DesAlt:NetCntxConcreteSuggestion );
DesAlt:NetCntxConcreteSuggestion = "*Focus on these substations:-";
SetSlotOption( DesAlt:NetCntxConcreteSuggestion, IF_NEEDED, NULL );
SetSlotOption( DesAlt:NetCntxConcreteSuggestion, WHEN_ACCESS,
NULL );
```

```
SetSlotOption( DesAlt:NetCntxConcreteSuggestion, BEFORE_CHANGE,
NULL );
SetSlotOption( DesAlt:NetCntxConcreteSuggestion, AFTER_CHANGE,
NULL );

MakeSlot( DesAlt:InstanceIdentifier );
SetSlotOption( DesAlt:InstanceIdentifier, IF_NEEDED, NULL );
SetSlotOption( DesAlt:InstanceIdentifier, WHEN_ACCESS, NULL );
SetSlotOption( DesAlt:InstanceIdentifier, BEFORE_CHANGE, NULL );
SetSlotOption( DesAlt:InstanceIdentifier, AFTER_CHANGE,
IdentifyToTrace );

MakeSlot( DesAlt:NetCntxAbstractArgument );
SetSlotOption( DesAlt:NetCntxAbstractArgument, MULTIPLE );
ClearList( DesAlt:NetCntxAbstractArgument );
SetSlotOption( DesAlt:NetCntxAbstractArgument, IF_NEEDED, NULL );
SetSlotOption( DesAlt:NetCntxAbstractArgument, WHEN_ACCESS, NULL
);
SetSlotOption( DesAlt:NetCntxAbstractArgument, BEFORE_CHANGE,
NULL );
SetSlotOption( DesAlt:NetCntxAbstractArgument, AFTER_CHANGE,
NULL );

    /*****************************************
     **** CLASS: LoadTransfer
     *****************************************/
MakeClass( LoadTransfer, DesAlt );


/************** METHOD: GetNetCntxConcrete  **************/

MakeMethod( LoadTransfer, GetNetCntxConcrete, [],
   {
    If Null?( GetValue( Self, ToSubStn ) )
     Then none
     Else GetValue( Self, ToSubStn );
   } );

/*************** METHOD: UpdateIdentifier  ***************/

MakeMethod( LoadTransfer, UpdateIdentifier, [],
If Null?( GetValue( Self, FromSubStn ) )
  Then ( Self:InstanceIdentifier = "Load transfer to " # GetValue( Self,
                                        ToSubStn ) )
 Else If Null?( GetValue( Self, ToSubStn ) )
       Then ( Self:InstanceIdentifier = "Load transfer from "
             # GetValue( Self, FromSubStn ) )
       Else Self:InstanceIdentifier = "From " # GetValue( Self,
                                        FromSubStn )
            # " To " # GetValue( Self, ToSubStn ) );
```

```
SetSlotOption( LoadTransfer:QualsSupporting, SINGLE );
LoadTransfer:QualsSupporting = ReduceEquip;
SetSlotOption( LoadTransfer:QualsCountering, SINGLE );
LoadTransfer:QualsCountering = SupplCloseLoad;
SetSlotOption( LoadTransfer:QualsRelevant, SINGLE );
LoadTransfer:QualsRelevant = MaxFlexibility;

MakeSlot( LoadTransfer:FromSubStn );
SetSlotOption( LoadTransfer:FromSubStn, IF_NEEDED, NULL );
SetSlotOption( LoadTransfer:FromSubStn, WHEN_ACCESS, NULL );
SetSlotOption( LoadTransfer:FromSubStn, BEFORE_CHANGE, NULL );
SetSlotOption( LoadTransfer:FromSubStn, AFTER_CHANGE,
UpdateIdentifier );

MakeSlot( LoadTransfer:ToSubStn );
SetSlotOption( LoadTransfer:ToSubStn, IF_NEEDED, NULL );
SetSlotOption( LoadTransfer:ToSubStn, WHEN_ACCESS, NULL );
SetSlotOption( LoadTransfer:ToSubStn, BEFORE_CHANGE, NULL );
SetSlotOption( LoadTransfer:ToSubStn, AFTER_CHANGE,
UpdateIdentifier );

MakeSlot( LoadTransfer:Voltage );
SetSlotOption( LoadTransfer:Voltage, ALLOWABLE_VALUES, primary,
secondary );
SetSlotOption( LoadTransfer:Voltage, IF_NEEDED, NULL );
SetSlotOption( LoadTransfer:Voltage, WHEN_ACCESS, NULL );
SetSlotOption( LoadTransfer:Voltage, BEFORE_CHANGE, NULL );
SetSlotOption( LoadTransfer:Voltage, AFTER_CHANGE, NULL );

MakeSlot( LoadTransfer:SupplyCondition );
SetSlotOption( LoadTransfer:SupplyCondition, ALLOWABLE_VALUES,
normal, outage );
LoadTransfer:SupplyCondition = normal;
SetSlotOption( LoadTransfer:SupplyCondition, PROMPT, "Will load be
transferred under normal or outage conditions?" );
SetSlotOption( LoadTransfer:SupplyCondition, IF_NEEDED, NULL );
SetSlotOption( LoadTransfer:SupplyCondition, WHEN_ACCESS, NULL );
SetSlotOption( LoadTransfer:SupplyCondition, BEFORE_CHANGE, NULL
);
SetSlotOption( LoadTransfer:SupplyCondition, AFTER_CHANGE, NULL );

MakeSlot( LoadTransfer:Quantity );
SetSlotOption( LoadTransfer:Quantity, VALUE_TYPE, NUMBER );
SetSlotOption( LoadTransfer:Quantity, MINIMUM_VALUE, 1 );
SetSlotOption( LoadTransfer:Quantity, MAXIMUM_VALUE, 250 );
SetSlotOption( LoadTransfer:Quantity, PROMPT, "What is the expected
load to be transferred (in MVA)?" );
SetSlotOption( LoadTransfer:Quantity, IF_NEEDED, NULL );
SetSlotOption( LoadTransfer:Quantity, WHEN_ACCESS, NULL );
SetSlotOption( LoadTransfer:Quantity, BEFORE_CHANGE, NULL );
SetSlotOption( LoadTransfer:Quantity, AFTER_CHANGE, NULL );
```

```
LoadTransfer:LongName = "Load Transfer";
SetValue( LoadTransfer:NetCntxAbstractTrigger, AdjZoneOfSupply,
Session8 );
SetValue( LoadTransfer:NetCntxAbstractSuggestion, "*Look at adjacent
zones", "*of supply for spare capacity." );
SetValue( LoadTransfer:NetCntxAbstractArgument, "Support for this
alternative may come from looking at", "zones of supply adjacent to the sub-
station to which", "load is being transferred to see if there is any spare
capacity." );

    /*************************************
    **** CLASS: NetReinforce
    *************************************/

MakeClass( NetReinforce, DesAlt );
NetReinforce:LongName = "Network Reinforcement";
NetReinforce:QualsCountering = ReduceEquip;
MakeSlot( NetReinforce:Location );
MakeSlot( NetReinforce:Connections );

    /*************************************
    **** CLASS: ExtendSS
    *************************************/

MakeClass( ExtendSS, NetReinforce );

/*************** METHOD: GetNetCntxConcrete  **************/

MakeMethod( ExtendSS, GetNetCntxConcrete, [],
  {
   If Null?( GetValue( Self, SubStnName ) )
    Then none
    Else GetValue( Self, SubStnName );
  } );

/*************** METHOD: UpdateIdentifier  **************/

MakeMethod( ExtendSS, UpdateIdentifier, [],
   Self:InstanceIdentifier = "Extension of  " # GetValue( Self,
                                SubStnName )
      # " substation" );
ExtendSS:LongName = "Extend (an existing) substation";
SetValue( ExtendSS:NetCntxAbstractTrigger, SubStnEquip, Session10 );
SetValue( ExtendSS:NetCntxAbstractSuggestion, "*Look at connections to
substation", "*which is to be extended and", "*plant currently installed." );
ExtendSS:QualsRelevant = SupplCloseLoad;
ExtendSS:QualsSupporting = SpreadLoadGrth;

MakeSlot( ExtendSS:SubStnName );
SetSlotOption( ExtendSS:SubStnName, IF_NEEDED, NULL );
SetSlotOption( ExtendSS:SubStnName, WHEN_ACCESS, NULL );
```

```
SetSlotOption( ExtendSS:SubStnName, BEFORE_CHANGE, NULL );
SetSlotOption( ExtendSS:SubStnName, AFTER_CHANGE,
UpdateIdentifier );
SetValue( ExtendSS:NetCntxAbstractArgument, "Consideration needs to
be given to what connection exists at the", "substation which is to be
extended and what plant is currently installed there." );

    /*************************************
    **** CLASS: EstablishSS
    *************************************/

MakeClass( EstablishSS, NetReinforce );

/*************** METHOD: UpdateIdentifier  ***************/

MakeMethod( EstablishSS, UpdateIdentifier, [],
   If Null?( GetValue( Self, Location ) )
     Then ( Self:InstanceIdentifier = " Substation connected at "
            # GetValue( Self, Connections ) )
     Else If Null?( GetValue( Self, Connections ) )
          Then ( Self:InstanceIdentifier = " Substation at "
                 # GetValue( Self, Location ) )
          Else Self:InstanceIdentifier = " Substation at " #
               GetValue( Self, Location ) # " connected at "
               # GetValue( Self, Connections ) );
EstablishSS:QualsSupporting = SupplCloseLoad;
EstablishSS:QualsRelevant = SpreadLoadGrth;
EstablishSS:LongName = "Establish a (new) substation";
SetSlotOption( EstablishSS:Location, AFTER_CHANGE, UpdateIdentifier );
SetSlotOption( EstablishSS:Connections, AFTER_CHANGE,
UpdateIdentifier );

    /*************************************
    **** CLASS: EquipReplmnt
    *************************************/

MakeClass( EquipReplmnt, DesAlt );
EquipReplmnt:LongName = "Equipment Replacement";

    /*************************************
    **** CLASS: NetUprate
    *************************************/

MakeClass( NetUprate, DesAlt );
NetUprate:LongName = "network Uprating";

    /*************************************
    **** CLASS: ResourceRlse
    *************************************/

MakeClass( ResourceRlse, DesAlt );
```

```
ResourceRlse:LongName = "Release of Resources";

    /*****************************************
     **** CLASS: NetCntx
     *****************************************/

MakeClass( NetCntx, Root );

    /*****************************************
     **** CLASS: ZoneOfSupply
     *****************************************/

MakeClass( ZoneOfSupply, NetCntx );

/*************** METHOD: CalcSpareCapacity   **************/

MakeMethod( ZoneOfSupply, CalcSpareCapacity, [SpareCapacity ],
   {
   Self:SpareCapacity = Self:InstalledCapacity * ( ( 100 -
PercentageFirmNormal  )
                                        / 100 );
   } );

/*************** METHOD: EstablishBoundary   **************/

MakeMethod( ZoneOfSupply, EstablishBoundary, [],
   NULL ); - not implemented
MakeSlot( ZoneOfSupply:InstalledCapacity );
MakeSlot( ZoneOfSupply:PercentageFirmNormal );
MakeSlot( ZoneOfSupply:PercentageFirmOutage );
MakeSlot( ZoneOfSupply:ProjectedGrowth );
MakeSlot( ZoneOfSupply:SpareCapacity );
SetSlotOption( ZoneOfSupply:SpareCapacity, IF_NEEDED,
CalcSpareCapacity );
SetSlotOption( ZoneOfSupply:SpareCapacity, WHEN_ACCESS, NULL );
SetSlotOption( ZoneOfSupply:SpareCapacity, BEFORE_CHANGE, NULL );
SetSlotOption( ZoneOfSupply:SpareCapacity, AFTER_CHANGE, NULL );

    /*****************************************
     **** CLASS: AdjZoneOfSupply
     *****************************************/

MakeClass( AdjZoneOfSupply, ZoneOfSupply );

    /*****************************************
     **** CLASS: SubStnZoneOfSupply
     *****************************************/

MakeClass( SubStnZoneOfSupply, ZoneOfSupply );
```

```
/************************************
 **** CLASS: PrimDistSys
 ************************************/

MakeClass( PrimDistSys, NetCntx );
MakeSlot( PrimDistSys:SuperGridSupplying );
SetSlotOption( PrimDistSys:SuperGridSupplying, IF_NEEDED, NULL );
SetSlotOption( PrimDistSys:SuperGridSupplying, WHEN_ACCESS, NULL
);
SetSlotOption( PrimDistSys:SuperGridSupplying, BEFORE_CHANGE,
NULL );
SetSlotOption( PrimDistSys:SuperGridSupplying, AFTER_CHANGE,
NULL );
MakeSlot( PrimDistSys:SubStnsSameSuperGrid );
SetSlotOption( PrimDistSys:SubStnsSameSuperGrid, MULTIPLE );
ClearList( PrimDistSys:SubStnsSameSuperGrid );
SetSlotOption( PrimDistSys:SubStnsSameSuperGrid, IF_NEEDED, NULL
);
SetSlotOption( PrimDistSys:SubStnsSameSuperGrid, WHEN_ACCESS,
NULL );
SetSlotOption( PrimDistSys:SubStnsSameSuperGrid, BEFORE_CHANGE,
NULL );
SetSlotOption( PrimDistSys:SubStnsSameSuperGrid, AFTER_CHANGE,
NULL );
MakeSlot( PrimDistSys:SuperGridInterconnection );

/************************************
 **** CLASS: SubStnEquip
 ************************************/

MakeClass( SubStnEquip, NetCntx );

/*************** METHOD: CalcSpareCapacity  **************/

MakeMethod( SubStnEquip, CalcSpareCapacity, [sparecapacity ],
  {
   Self:SpareCapacity = Self:InstalledCapacity * ( ( 100 -
PercentageFirmNormal )
                                    / 100 );
  } );
MakeSlot( SubStnEquip:SpareCapacity );
SetSlotOption( SubStnEquip:SpareCapacity, IF_NEEDED,
CalcSpareCapacity );
SetSlotOption( SubStnEquip:SpareCapacity, WHEN_ACCESS, NULL );
SetSlotOption( SubStnEquip:SpareCapacity, BEFORE_CHANGE, NULL );
SetSlotOption( SubStnEquip:SpareCapacity, AFTER_CHANGE, NULL );

MakeSlot( SubStnEquip:PercentageFirmNormal );
MakeSlot( SubStnEquip:PercentageFirmOutage );
MakeSlot( SubStnEquip:ProjectedGrowth );
MakeSlot( SubStnEquip:InstalledCapacity );
```

```
/*****************************************
 ****  CLASS: Plant
 *****************************************/

MakeClass( Plant, SubStnEquip );

/*************** METHOD: CalcUsefulLife  ***************/

MakeMethod( Plant, CalcUsefulLife, [UsefulLife ],
  {
   Self:UsefulLife = NormalLife - Age;
  } );

MakeSlot( Plant:NormalLife );
MakeSlot( Plant:Age );
MakeSlot( Plant:PastPerformance );
MakeSlot( Plant:LoadingHistory );
MakeSlot( Plant:UsefulLife );
SetSlotOption( Plant:UsefulLife, IF_NEEDED, CalcUsefulLife );
SetSlotOption( Plant:UsefulLife, WHEN_ACCESS, NULL );
SetSlotOption( Plant:UsefulLife, BEFORE_CHANGE, NULL );
SetSlotOption( Plant:UsefulLife, AFTER_CHANGE, NULL );

/*****************************************
 ****  CLASS: Switchgear
 *****************************************/

MakeClass( Switchgear, Plant );

/*****************************************
 ****  CLASS: Transformers
 *****************************************/

MakeClass( Transformers, Plant );

/*****************************************
 ****  CLASS: CablesIncoming
 *****************************************/

MakeClass( CablesIncoming, Plant );

/*****************************************
 ****  CLASS: CablesOutgoing
 *****************************************/

MakeClass( CablesOutgoing, Plant );
```

```
/*************************************
 **** CLASS: SensitiveLoad
 *************************************/

MakeClass( SensitiveLoad, NetCntx );

/*************************************
 **** CLASS: DesignLog
 *************************************/

MakeClass( DesignLog, Root );

/************** METHOD: SummarizeAllAlts  **************/

MakeMethod( DesignLog, SummarizeAllAlts, [],
  {
   DisplayText( Transcript5, "Summary of alternatives generated in
chronological  sequence:",
      FormatValue( "\n" ) );
   EnumList( Self:AllInstancesList, x,
    {
     DisplayText( Transcript5, x, "  ", x:LongName, " - ",
x:InstanceIdentifier,
        FormatValue( "\n" ) );
    });
  });

/*************** METHOD: GetAllInstancesList  ***************/

MakeMethod( DesignLog, GetAllInstancesList, [],
   GetValue( Global, Instlist ) );


/*************** METHOD: GetAllClassesList  ***************/

MakeMethod( DesignLog, GetAllClassesList, [],
  {
   ResetValue( Global:tList );
   EnumList( Self:AllInstancesList, x,
    {
    If Member?( Global:tList, GetParent( x ) )
     Then NULL
     Else AppendToList( Global:tList, GetParent( x ) );
   });
GetValue( Global, tList );
});
```

```
/*************** METHOD: GetSpecificClassInstList   ***************/

MakeMethod( DesignLog, GetSpecificClassInstList, [classname ],
  {
   ResetValue( Global:tList );
   EnumList( Self:AllInstancesList, x,
    {
     If IsAKindOf?( x, classname )
      Then AppendToList( Global:tList, x )
      Else NULL;
    } );
   GetValue( Global, tList );
  } );


/*************** METHOD: SummarizeClassOfAlts   ***************/

MakeMethod( DesignLog, SummarizeClassOfAlts, [],
  {
   Let [reply PostMenu( "Choose type of alternative to review",
            GetValue( Self, AllClassesList ) )]
    {
     DisplayText( Transcript5, "Summary of alternatives generated of
specified type:",
       FormatValue( "\n" ) );
     EnumList( SendMessage( Self, GetSpecificClassInstList,
            reply ), x,
      {
       DisplayText( Transcript5, x, "  ", x:LongName, " - ",
x:InstanceIdentifier,
         FormatValue( "\n" ) );
      } );
    };
  } );


/*************** METHOD: ShowArgumentforaDesAlt   ***************/

MakeMethod( DesignLog, ShowArgumentforaDesAlt, [],
  {
   Let [reply PostMenu( "Choose alternative to review",
            GetValue( Self, AllInstancesList ) )]
    {
     DisplayText( Transcript5, "Summary of argument related to selected
alternative:",
       FormatValue( "\n" ) );
     DisplayText( Transcript5, reply:LongName, " - ",
reply:InstanceIdentifier,
       FormatValue( "\n" ) );
     DisplayText( Transcript5, "    Supporting design commitments - ",
       FormatValue( "\n" ) );
```

```
        EnumList( GetValue( GetValue( reply, QualsSupporting ),
                ArgumentSummary ), x, DisplayText( Transcript5,
                        x, FormatValue( "\n" ) ) );
        DisplayText( Transcript5, "    Non-supporting design commitments - ",
          FormatValue( "\n" ) );
        EnumList( GetValue( GetValue( reply, QualsCountering ),
                ArgumentSummary ), x, DisplayText( Transcript5,
                        x, FormatValue( "\n" ) ) );
        DisplayText( Transcript5, "    Other relevant design commitments - ",
          FormatValue( "\n" ) );
        EnumList( GetValue( GetValue( reply, QualsRelevant ),
ArgumentSummary  ),
          x, DisplayText( Transcript5, x, FormatValue( "\n" ) ) );
       DisplayText( Transcript5, "Relevant current elements of the electrical
network :",
          FormatValue( "\n" ) );
       EnumList( GetValue( reply, NetCntxAbstractArgument ), x,
          DisplayText( Transcript5, x, FormatValue( "\n" ) ) );
       DisplayText( Transcript5, "    (", reply:NetCntxConcreteSuggestion,
          reply:NetCntxConcrete, ") " );
    };
  } );


MakeSlot( DesignLog:AllInstancesList );
SetSlotOption( DesignLog:AllInstancesList, MULTIPLE );
ClearList( DesignLog:AllInstancesList );
SetSlotOption( DesignLog:AllInstancesList, IF_NEEDED,
GetAllInstancesList );
SetSlotOption( DesignLog:AllInstancesList, WHEN_ACCESS, NULL );
SetSlotOption( DesignLog:AllInstancesList, BEFORE_CHANGE, NULL );
SetSlotOption( DesignLog:AllInstancesList, AFTER_CHANGE, NULL );

MakeSlot( DesignLog:AllClassesList );
SetSlotOption( DesignLog:AllClassesList, MULTIPLE );
ClearList( DesignLog:AllClassesList );
SetSlotOption( DesignLog:AllClassesList, IF_NEEDED, GetAllClassesList
);
SetSlotOption( DesignLog:AllClassesList, WHEN_ACCESS, NULL );
SetSlotOption( DesignLog:AllClassesList, BEFORE_CHANGE, NULL );
SetSlotOption( DesignLog:AllClassesList, AFTER_CHANGE, NULL );

/*****************************************************/
        ALL GLOBAL INSTANCES ARE SHOWN BELOW
/*****************************************************/
MakeSlot( Global:Instlist );
SetSlotOption( Global:Instlist, INHERIT, FALSE );
SetSlotOption( Global:Instlist, MULTIPLE );
SetSlotOption( Global:Instlist, VALUE_TYPE, OBJECT );
```

```
MakeSlot( Global:NewestInstance );
SetSlotOption( Global:NewestInstance, INHERIT, FALSE );

MakeSlot( Global:tClass );
SetSlotOption( Global:tClass, INHERIT, FALSE );

MakeSlot( Global:tList );
SetSlotOption( Global:tList, INHERIT, FALSE );
SetSlotOption( Global:tList, MULTIPLE );
SetValue( Global:tList, LoadTransfer1, LoadTransfer2 );

MakeSlot( Global:SSContext );
SetSlotOption( Global:SSContext, IMAGE, Edit5 );
MakeSlot( Global:NewestDesAltClass );
SetSlotOption( Global:NewestDesAltClass, INHERIT, FALSE );

MakeSlot( Global:TraceIndentation );
SetSlotOption( Global:TraceIndentation, INHERIT, FALSE );
SetSlotOption( Global:TraceIndentation, VALUE_TYPE, NUMBER );
SetSlotOption( Global:TraceIndentation, MINIMUM_VALUE, 0 );
SetSlotOption( Global:TraceIndentation, MAXIMUM_VALUE, 20 );
Global:TraceIndentation = 0;


/*************************************************/
        FUNCTIONS ARE SHOWN BELOW
/*************************************************/

      /*****************************************
       **** FUNCTION: ShowCounteringQs
       *****************************************/
MakeFunction( ShowCounteringQs, [image],
   SendMessage( image, CounteringQs ) );

      /*****************************************
       **** FUNCTION: ShowRelevantQs
       *****************************************/
MakeFunction( ShowRelevantQs, [image],
   SendMessage( image, RelevantQs ) );

      /*****************************************
       **** FUNCTION: SetUp
       *****************************************/
MakeFunction( SetUp, [image],
   SendMessage( image, SetUp ) );

      /*****************************************
       **** FUNCTION: ShowSupportQs
       *****************************************/
MakeFunction( ShowSupportQs, [image],
```

```
SendMessage( image, SupportingQs ) );

/*************************************
 **** FUNCTION: ShowMoreGQualities
 *************************************/
MakeFunction( ShowMoreGQualities, [image],
   SendMessage( image, ShowParent ) );


/*************************************
 **** FUNCTION: ShowMoreSQualities
 *************************************/
MakeFunction( ShowMoreSQualities, [image],
   SendMessage( image, ShowChildren ) );

/*************************************
 **** FUNCTION: ShowAltQualities
 *************************************/
MakeFunction( ShowAltQualities, [image],
   SendMessage( image, ShowSiblings ) );

/*************************************
 **** FUNCTION: TidyUp
 *************************************/
MakeFunction( TidyUp, [image],
   SendMessage( image, TidyUp ) );

/*************************************
 **** FUNCTION: ShowNetCntxWin
 *************************************/
MakeFunction( ShowNetCntxWin, [],
  {
   ShowWindow( Session7 );
   If Null?( Global:NewestDesAltClass )
    Then {
        SendMessage( DesAlt, DispNetCntxAbstractSuggestion );
        SendMessage( DesAlt, DispNetCntxConcreteSuggestion );
       }
    Else {
        If Null?( Global:NewestInstance )
         Then {
             SendMessage( GetValue( Global, NewestDesAltClass ),
               DispNetCntxAbstractSuggestion );
  }
Else {
    If ( GetParent( Global:NewestInstance )
          #= Global:NewestDesAltClass )
     Then {
         SendMessage( GetValue( Global, NewestInstance ),
           DispNetCntxAbstractSuggestion );
         SendMessage( GetValue( Global, NewestInstance ),
```

```
                DispNetCntxConcreteSuggestion );
            }
          Else{
              SendMessage( GetValue( Global, NewestDesAltClass ),
                DispNetCntxAbstractSuggestion );
            };
        };
    };
} );

  /*****************************************
   **** FUNCTION: ShowTraceWin
   *****************************************/
MakeFunction( ShowTraceWin, [],
   ShowWindow( Session12 ) );

    /*****************************************
     **** FUNCTION: HideTraceWin
     *****************************************/
MakeFunction( HideTraceWin, [],
   HideWindow( Session12 ) );

    /*****************************************
     **** FUNCTION: IncreaseTraceIndentation
     *****************************************/
MakeFunction( IncreaseTraceIndentation, [],
   Global:TraceIndentation = Global:TraceIndentation + 1 );

    /*****************************************
     **** FUNCTION: DecreaseTraceIndentation
     *****************************************/
MakeFunction( DecreaseTraceIndentation, [],
   Global:TraceIndentation = Global:TraceIndentation - 1 );

    /*****************************************
     **** FUNCTION: IndentTrace
     *****************************************/
MakeFunction( IndentTrace, [],
   For x From 1 To GetValue( Global, TraceIndentation )
     Do DisplayText( Transcript4, "  >" ) );

/*****************************************************************************/
```

APPENDIX 6

Demonstrator's Script

(1) Introduction

1. Load KAPPA-PC and Open PDP application.
2. Describe KAPPA-PC developers' interface.

(2) Statics: Overview of Object Hierarchies and Orthogonal Reasoning Using Message Passing

1. Maximise Object Hierarchy.
2. Briefly describe the four object class hierarchies using figure of Design Situation to show parts which have been implemented. Point out that display gives a STATIC representation of object hierarchies only (orthogonal links - i.e. links between hierarchies are not shown.)
3. Mention the (usual) inheritance mechanisms (for generalization and specialization) which are supported by arranging object classes in hierarchies.
4. Explain how orthogonal reasoning is effected generally by message passing between methods in (different) object hierarchies.
5. Explain how design alternatives are associated with qualitative commitments, referring to supporting, counter indicating and relevant associations. Use display of object hierarchy to do this.
6. Show examples of slots and methods (using DesAlt), mentioning that behaviour of objects is inherited whenever possible.
7. Show DesAlt methods SupportingQs, CounteringQs and RelevantQs which use slots (in DesAlt) QualsSupporting, QualsCountering and QualsRelevant respectively to send messages to the method Display Self in the appropriate object (class) in the QualsComm hierarchy. Explain that this mechanism activates orthogonal reasoning; namely, supporting, counter indicated and other relevant qualitative design commitments are related to design activity (objects in the DesAlt hierarchy) by message passing prompted by the design alternative being pursued by the designer.
8. Resize Object Hierarchy.

(3) Dynamics: Supporting the designer's consideration of a design alternative by showing associated design qualities.

1. Invoke PDP Session. Explain interface rationale - essentially a developer's interface which enables the dynamics supported by the system's architecture to be demonstrated.
2. Explain that when a designer is considering a particular design alternative he can see which qualitative commitments

are associated with the possibility he is considering - which qualities support the design alternative, which are counter indicated by it, and which others are relevant in some way i.e. how design alternatives are associated with the qualitative commitments (hierarchy of object classes) in a focused fashion.

3. Click on "Explore Alternatives" button.
4. Click on "Load Transfer" button, and explain transferring load as a design option.
5. Show link to evaluation of qualities by clicking on "Supporting" button.
    1. Describe and demonstrate movement around qualitative commitments taxonomy from the starting focus using displayed buttons and figure of QualComms hierarchy.
    2. Return to Load Transfer window.
6. Demonstrate other forms of the by clicking on "Countering" or "Relevant" button, or iterate using each in turn.
7. Return to Load Transfer window.
8. Explain how a designer might be presented with qualities for inclusion in a text to justify a load transfer proposal.

[Prepare for demonstration of Design Log by the following -
1. Click on "Enter Load Transfer Details" button.
2. Enter token data to represent a load transfer.
3. Return to Load Transfer window.
4. Return to Explore Alternatives window.]

... optionally give another example of orthogonal link from DesAlt hierarchy to QualComm hierarchy ...

9. Explain network reinforcement as design options.
10. Click on "New S/S" button.
11. Show (different examples of) links to design qualities by clicking on "Supporting", "Relevant" or "Countering" button (or iterate with a selection). Use figure of QualComms hierarchy to show where focus starts and how exploration fans out.
12. Return to New S/S window.

[Prepare for demonstration of Design Log by the following -
1. Click on "Enter New S/S Details".
2. Enter token data to represent a new substation.
3. Return to New S/S window.
4. Return to Explore Alternatives window.]

13. Return through the window(s) to the PDP top level.

(4) Dynamics: Supporting the designer in pursuing a design alternative by responding to what appears to be relevant in the electrical network context by suggesting abstract and concrete foci.

1. Explain how design alternatives are associated with the electrical network context introducing the notion of abstract and concrete foci.
2. At the PDP top level window click the "Review Electrical Network" button. Explain access to electrical network context by specifying substation name of interest (which would give the concrete focus) and the selection of an aspect of the electrical network whichis of interest (abstract focus). Point out that as no design alternative is currently being pursued the system cannot suggest any sort of focus.
3. Return from reviewing the electrical network and click on "Explore Alternatives" button.
4. Click on "Load Transfer" as type of alternative to consider.
5. Click on "Review Electrical Network" button from Load Transfer window.
6. Point out the abstract context suggested by the system based on its assumption that the designer may be interested in aspects of the electrical network associated with transferring load. Click on "Follow Up Suggestion" to demonstrate correct link is being made. Point out that the designer is free to follow up the system's suggestion or to review any aspect of the electrical network context of his own choosing.
7. Return from reviewing the electrical network and click on the "Enter Load Transfer Details" button.
8. Enter token load transfer details, explain that for load transfers the designer willbe interested in zones of supply adjacent to the substation receiving the load transfer.
9. Click on the "Review Electrical Network" button from the screen where load transfer details have been entered.
10. Point out the abstract context suggested by the system based on its assumption that the designer may be interested in aspects of the electrical network associated with transferring load - as before. Point out that this time the system is able to qualify its abstract suggestion with a concrete focus since details of a specific load transfer have been entered by the designer. Click on "Follow Up Suggestion" to demonstrate correct link is being made, pointing out that since no plant and circuits data(base) is linked to this system the concrete focus does not affect what is actually displayed here.

... optionally repeat part of the demonstration using the design alternative of extending a substation to demonstrate different abstract focus and use of a different attribute of the specific instance of a design alternative (in this case the name of the substation to be extended) ...

(5) Dynamics: Design Log (and Trace)

1. Draw attention to the small part of a trace visible at the foot of the screen.
2. Click on "Browse Trace" button.

3. Describe trace - reviewing the activities in the demonstration so far, it gives a sequential record of the dynamics of this session. Explain the role of this "canned text" trace which is NOT intended to be used by a designer/user - role is entirely as a resource for reflection on the demonstration - as it has just been used.
4. Click on "Design Log" button.
5. Explain the options and click on the button to show design alternatives developed so far (in chronological order). Explain how this display is produced by using a list of design alternatives explored and retrieving for each entry in the list information to be displayed by sending messages from the Design Log (class of) object to each of the object instances which have been created under the appropriate classes of Design Alternatives.
6. Click on the button to show instances of specified class of design alternatives, continuing explanation.
7. Click on the button to summarize the argument for a specific design alternative.
8. Explain how design commitments and relevant electrical network context are linked via the instance of a design alternative.
9. Click on the "Return" button.
10. Close the PDP SESSION window.(Do not click on "Exit"- by closing the window the instances of design alternatives which have been created will be retained, whereas exiting resets the system to its start up state.)
11. Maximize the Object Hierarchy to show the instances of design alternatives which have been created and the Design Log object class.
12. Show Design Log method ShowArgumentforaDesAlt which sends a message to the appropriate instance of a class of design alternative - instances now have appeared as leaves in the DesAlt object class hierarchy. Explain how the Design Log object makes use of the design alternative instances to make the design support given open to inspection.

The structures which have been implemented although they are instantiated rather skeletally in the demonstration can be scaled up and hence the system demonstrated constitutes a proof of concept.

The architecture which supports the behaviour demonstrated can handle expansion in terms of both volume and variety. The design log supports the giving of a logical justification for the design activity.

The system has the structure that it has because of the need for explanation and justification which demands that the knowledge representation be explicit (hence the system's static qualities) and that the reasoning be open to inspection (hence the system's means of representing and recording the dynamic behaviour).