Illustration of immature HCI engineering:

Carry Forward in the Development of Military Planning Systems

Martin Colbert

University College London

1994

Thesis submitted for the Degree of Doctor of Philosophy in the Faculty of Engineering, University of London

Acknowledgement

special mention: John Long (supervisor)

mention: all my colleagues at Ergonomics & HCI Unit;
Chris Young and Chris Madams (colleagues at DRA Portsdown);
Carol Gaye, DRA, and Brian Drabble, AIAI (demonstration and discussion of actual planning systems);

dedication: to my wife Lyn

Abstract

In recent years, a number of difficulties in designing interactions between military personnel and their command and control (C^2) systems have been identified. These difficulties are persistent and have been attributed to a lack of carry forward between procurement projects.

In response to these difficulties, this thesis attempts to realise and then illustrate carry forward in a manner that is characteristic of a particular form of the discipline of Human Computer Interaction (HCI) - informal HCI Engineering. In essence, informal HCI Engineering is different from current best practice in that design work addresses *general classes* of design Problem and *instances of general classes* of design Problem, rather than just problem instances which are related to other instances in some, unspecified way. Consequently, in principle, informal Engineering offers *additional* opportunities to develop and apply knowledge to design work. Specifically, it offers additional opportunities to support: (i) the abstraction of general requirements from instance requirements; (ii) the production of general specifications in response to general requirements; and (iii) the instantiation of general specifications for particular instances. Further, the knowledge applied in support of design may concern *classes* of design Problem, rather than just instances or a poorly specified range of instances. In addition, informal Engineering provides an *additional* way of reasoning about the completeness and/or selectivity with which design Problem instances are addressed - reasoning with respect to a relevant class.

In this thesis, carry forward in the desired manner is enabled by acquiring the minimum amount of knowledge necessary for carry forward of some kind - a preliminary conception of the domain of C^2. Carry forward is then realised by using this preliminary conception to evaluate and specify selected aspects of military planning systems reconstructed in a laboratory for purposes of research. To highlight the distinctive characteristics of carry forward in informal HCI Engineering, and to monitor its potential effectiveness in practice, each attempt to realise carry forward is compared to the current best practice equivalent.

Two attempts are made to realise carry forward in the desired manner, first, in late evaluation, and second, in specification. Of these, the second attempt is judged to be more satisfying than the first, since: (i) carry forward is fully carried through; (ii) both a general requirement and a general specification are developed; and (iii) in the case reported, the value of the specifications produced are judged likely to outweigh the costs of their development.

Future work may seek to scale-up and transfer to actual design Problems, the manner of carry forward illustrated here.

Contents

Chapter 2. A Preliminary Conception of the Domain of Military Command and Control

Chapter 3. First Attempt: Carry Forward in Evaluation

Figures

Tables

Chapter 1.
Forms of HCI and the Zero-->Build Strategy

Summary

In recent years, a number of difficulties designing interactions between military personnel and their command and control (C^2) systems have been identified. These difficulties are persistent and have been attributed to a lack of carry forward between procurement projects.

The discipline which must respond to these difficulties by achieving greater carry forward is Human-Computer Interaction (HCI). A discipline, here, is taken to comprise knowledge supporting practices addressing problems. Within this view of a discipline, carry forward may be initially conceived as the *realisation* of a combination of knowledge, practices and problems, that is, an achievement of knowledge that in actual fact supports a number of practices and so in actual fact addresses a number of problems. The notions of discipline and carry forward, then, are closely related (since a discipline which has yet to realise an effective combination of knowledge, practices, and problems has little basis for claims of discipline status). Consequently, an analysis of different forms of HCI, that is, different combinations of knowledge, practices and problems may come to reveal how carry forward may be achieved in different manners. Such an analysis may also serve to explicate the nature of the difficulties with the procurement of C^2 systems and suggest how HCI as a discipline may respond.

In this thesis, it is suggested that lack of carry forward is an integral part of one particular form of HCI - implicit HCI Craft. Lack of carry forward is simply a corollary of designing in a certain way. As a matter of fact, rather than principle, lack of carry forward tends to be related to, and to go together with, 'soft' design problems, personal skills and expertise and iterative implementation and testing. In recent years, many interactions between military personnel and their C^2 systems have been designed in just such a manner.

Since lack of carry forward is an integral part of implicit Craft, attempts to achieve greater carry forward must seek to progress HCI as a discipline.

Greater carry forward requires the construction of a fundamentally different and more effective combination of knowledge, practices and problems. There is a need to reconstruct *all* the elements of the discipline *together*, to devise knowledge representations that enable carry forward, a way of designing with knowledge so represented, and design problems that may be effectively addressed by this knowledge and in this way. Attempts to modify a single element of the discipline in isolation of the others are unlikely to be effective, because comprehension of the relationships between knowledge, practices and problems is currently too low to predict, first, whether a single discipline element (to be developed now) will indeed complement other necessary elements (to be developed later), and, second, whether a particular combination, if ever realised, will indeed be effective in practice.

Before seeking to construct an additional form of HCI, it is necessary to make clear that attempts to progress HCI as a discipline may not assuage all of the difficulties with procurement all of the time. If a 'more advanced' form of HCI is developed, then this 'more advanced' form may be unable to address *all* the C^2 design problems that are currently addressed by implicit Craft. Some C^2 problems may be so 'soft' and poorly specified that hand-crafting may be the most effective manner of address, and so lack of carry forward may just have to be accepted and accommodated. Consequently, it may be unrealistic to expect greater carry forward to be universally achievable. However, progress at the discipline level may be expected to assuage at least *some* difficulties with procurement some of the time. Part of progressing HCI is the selection of 'harder', better-specified design problems that may be effectively addressed by 'more advanced' knowledge and practices, and the directing of 'more advanced' HCI towards these problems.

In this thesis, greater carry forward is sought through a form of HCI termed informal HCI Engineering. Informal HCI Engineering is different from current best practice in that design work addresses *general classes* of design Problem and *instances of general classes* of design Problem[1], rather than just problem instances which are related to other instances in some, unspecified way. Consequently, informal Engineering is of interest with

---

[1]The distinction between Problem (capital 'P') and problem (lower case 'p') is elaborated later (see sections 1.1.2. and 1.1.4.1.).

respect to carry forward since, in principle, it offers *additional* opportunities to develop and apply knowledge to design. Specifically, it offers additional opportunities to support: (i) the abstraction of general requirements from instance requirements; (ii) the production of general specifications in response to general requirements; and (iii) the instantiation of general specifications for particular instances. Further, the knowledge applied in support of design may concern *classes* of design Problem, rather than just instances or a poorly specified range of instances. In addition, informal Engineering also provides an *additional* way of reasoning about the completeness and/or selectivity with which design Problem instances are addressed - reasoning with respect to a relevant class.

The strategy for the development of informal HCI Engineering is 'Zero-->build'. That is, an attempt is made to develop knowledge about *classes* of Problem and apply such knowledge to general requirements and general specifications immediately and from nothing (or near nothing)[2]. The Zero-->build strategy is adopted to ensure that the knowledge acquired (informal models of C^2) indeed supports the desired practices (informal abstraction, specification, specialisation, instantiation and evaluation) which indeed address the desired problems (general Problems concerning military personnel interacting with C^2 computer equipment to manage armed-conflicts effectively). The Zero-->build strategy is also adopted to ensure that the Problems selected (aspects of C^2) are indeed sufficiently hard to be effectively addressed by informal abstraction and specialisation etc. In addition, the Zero-->build strategy is adopted to enable refinement of the thesis' objectives. Since the desired manner of carry forward is to be realised from near nothing ('zero'), the initial specification of the form of HCI to be realised is itself inadequate. Initial realisations of combinations of knowledge, practices and problems ('builds') may be compared and contrasted with current best, but carry-forward-deficient practice, in order to reveal more precisely the characteristics of the 'more advanced' form of HCI that is sought. With current knowledge, neither the compatibility of knowledge - practice - problem relations, nor the hardness of design concerns may be established *a priori*. The Zero-->build strategy recognises that, for the moment, an appropriate combination of knowledge,

---

[2] 'Zero-->build' as used in this thesis should not be confused with the term 'zero-build', which is sometimes used to refer to version 0.0 of a system.

practices and problems must be largely selected on a trial and error basis, or, at best, rules of thumb.

In essence, Zero-->build provides an opportunity to consider immediately, and on the basis of case evidence, the potential effectiveness *in practice* of the means of carry forward envisaged. Importantly, such considerations are not simply deferred, or reduced to consideration of effectiveness in principle. Zero-->build ensures that the work proceeds breadth-first and realises knowledge, problems and practices *in combination.* Given the inadequacy of current understanding of the relations between HCI knowledge, practices and problems, an attempt to develop one of these elements in depth, but in isolation from the others, is fraught with difficulty. With current knowledge, it is difficult to predict whether a single discipline element (to be developed now) will indeed complement other necessary elements (to be developed later) and that the combination will indeed be effective in practice. For example, it is difficult to predict the usability of certain knowledge representations in advance of tests with the intended designer population (Bellotti, 1988) or the efficacy of such representations in advance.

A consequence of adopting the Zero-->build strategy with the limited resources of a PhD is that, for the moment at least, it is only possible to address *selected* aspects of design Problems and to develop minimal knowledge representations with which to support design work. Consequently, to maximise the opportunity to practise effectively, any knowledge must focus upon the most informative and easily conceived aspects of C^2, and it is necessary, for the moment at least, to work with knowledge of a preliminary kind and selective address of design Problems. Also, the individual who develops the knowledge representations (the 'researcher') must also conduct the practice (be the 'designer'). Failure to do so may result in precious resources being wasted making explicit the tacit understanding of the knowledge acquired and disseminating this understanding to 'designers', assuming, of course, that such understanding may be made explicit at this stage. An additional consequence of pursuing the Zero-->build strategy with limited resources is that only a limited number of attempts to realise the desired manner of carry forward may be made. Consequently, to facilitate such attempts, the design problems addressed must be reconstructed problems and addressed in a laboratory context, rather than actual problems addressed in a commercial context. Finally, given limited resources, the effectiveness in practice of the means of carry forward realised must be assessed in a cost effective manner. Thus, for the moment at least, it is necessary to work with judgements and suggestive empirical reports, rather than allocate resources to more conclusive (but premature) evaluation methods.

Of course, the adoption of the Zero-->build strategy now does not preclude the adoption of other strategies later. For example, once knowledge - practice - problem relations are better comprehended and the nature of problem 'hardness' are better specified, a more 'top-down, depth-first' strategy more conventional in academic research may be more appropriate.

With the limitations of the Zero-->build strategy in mind, this thesis develops a preliminary conception of the domain of C^2 (hence pre-conception). A conception, here, is taken to be a set of concepts for formulating design Problems. The pre-conception is used to support two attempts to realise carry forward in a manner characteristic of informal HCI Engineering. The first attempt involves the evaluation of University College London's (UCL's) prototype system for planning the off-load of men and equipment during hypothetical amphibious operations (OPS1). The second attempt involves the specification of a partial menu structure for military planning systems generally, and its instantiation for OPS1 and a prototype military satellite construction scheduler (SATCON1), also reconstructed at UCL. For each attempt at carry forward, the work conducted is reported as an illustrative case history. (The case histories illustrate key characteristics of the manner of carry forward sought). Each actual case history (of work conducted in an informal, Engineering manner) is juxtaposed with an hypothetical case history (of how equivalent work could have been conducted in a more conventional, 'current best practice' manner). Finally, each actual case history is reviewed with respect to its effectiveness in practice, relative to the hypothetical alternative. The adequacy with which each case history illustrates carry forward in a manner characteristic of informal HCI Engineering is also considered.

There is little hope that the Zero>build strategy will realise the perfect form of informal HCI Engineering at the first attempt, if for no other reason than the nature of the objective is poorly specified at the outset. Indeed, as later chapters reveal, the first attempt appears to fail, and only at the second attempt is anything like informal HCI Engineering realised. However, zero-build may usefully demonstrate the feasibility and potential of this form of HCI with particular reference to carry forward.

By virtue of its preliminary, selective and reconstructive nature, then, this thesis may genuinely attempt to specify, realise and illustrate informal HCI Engineering immediately. This thesis reflects an exploratory exercise which, refining its objectives as it proceeds, seeks to advance HCI knowledge, practices and problems together by illustrating a combination of these elements that, judging from the case evidence, appears to be effective in practice. The inevitable limitation of the Zero-->build strategy is that none of the individual elements are realised in a highly selective way. For example, an alternative thesis, following a more 'top-down, depth-first' strategy, could have developed a model of C^2 more thoroughly, and sought to verify its scope and content, and assumed that other work would subsequently construct ways of using the model and identify problems to address with it. This thesis makes no argument against the adoption of such an alternative strategy, once more is known about knowledge - practice - problem relations and the nature of problem 'hardness'. Rather than wait until such knowledge is available, however, (assuming that somebody intends to develop it), this thesis argues for the necessity of the Zero-->build strategy, limitations included.

1.1. Different Forms of the Discipline of Human-Computer Interaction

1.1.1 Difficulties in Designing C^2 Systems

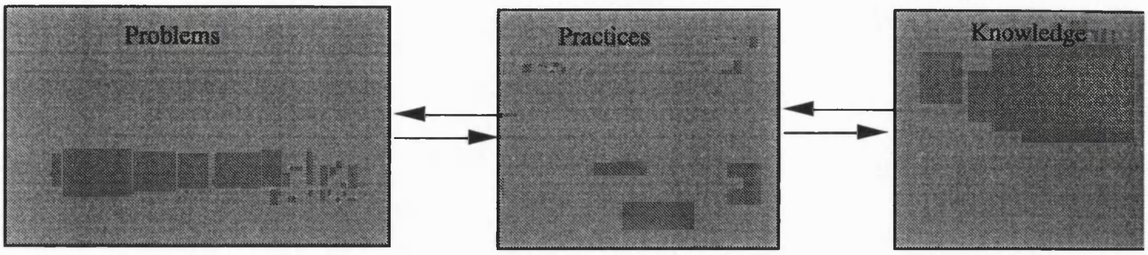
A number of difficulties in designing human-computer interactions for C^2 systems have been identified (Holman, Young & Dorrington, 1987). These difficulties include: (i) re-design during and after installation; (ii) cost escalation; (iii) built-in obsolescence; and (iv) project slip. Given the current concern with the level of defence spending and the changing nature of military commitments, such problems are pressing. However, they are not new. Indeed, such problems arise all too often and appear resistant to remedial action.

The persistence of such problems has been attributed to a lack of 'carry forward' (Jordan, Lee & Cawsey, 1988; Grundy, 1988). That is, in lay terms, despite the best efforts of those involved, the development and application of knowledge is inadequate - procurement projects are failing to learn enough from previous research or development work. To develop the technical view of carry forward necessary for this thesis, it is necessary to consider the nature of the discipline which must respond to these difficulties with procurement - Human-Computer Interaction (HCI).

1.1.2. Carry Forward and HCI as a Discipline

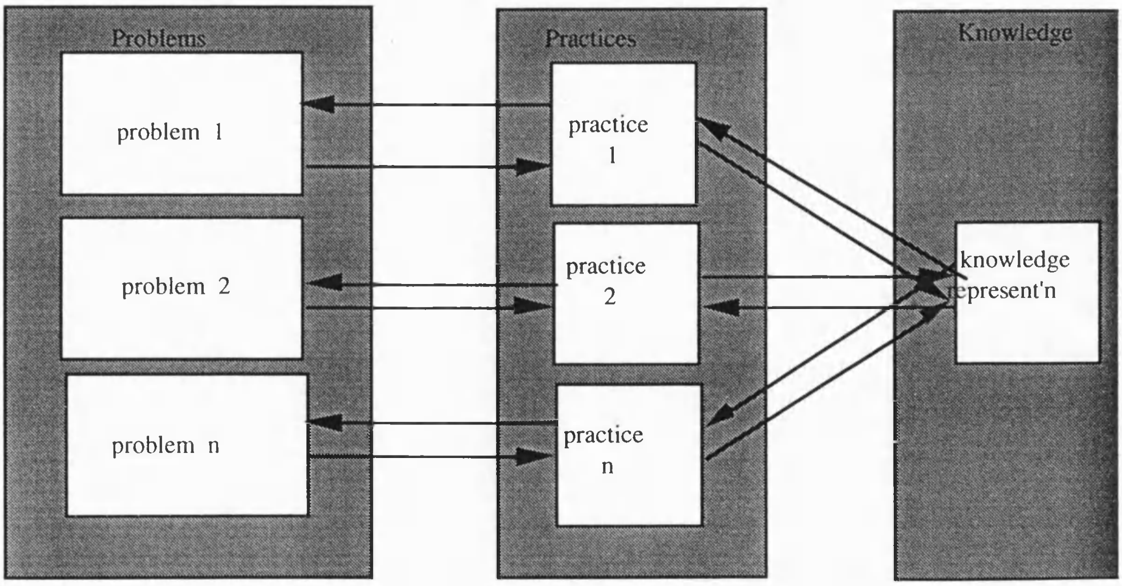
A discipline, here, is taken to comprise knowledge supporting practices addressing problems, and discipline knowledge may be methodological (how to design) or substantive (what to design)(Long & Dowell, 1989) (see Figure 1.1(i)). A practice, here, is taken to be a way for practitioners to behave that is acceptable to the discipline community. HCI, here, is taken to be a discipline that may take many forms and which addresses design problems involving humans interacting with computers to perform work effectively.

Figure 1.1(i): The Nature of a Discipline (after Long & Dowell, 1989)



Since the notion of carry forward focuses upon the relations between the principal elements of a discipline - knowledge, practices and problems - this characterisation of HCI as a discipline enables an initial technical view of carry forward as the *realisation* of a combination of knowledge, practices and problems, that is, an achievement of knowledge that in fact supports a number of practices, and so addresses a number of problems (see Figure 1.1(ii). The notions of discipline and carry forward, then, are closely related, since a discipline which has yet to realise an effective combination of knowledge, practices, and problems, and represented knowledge such that it may be re-used, has little basis for claims of discipline status.

Figure 1.1(ii): Carry Forward in HCI



Given this close relationship between carry forward and the nature of HCI as a discipline, it is important to be precise about the nature of HCI. This thesis characterised HCI as addressing "*design problems* involving humans interacting with computers to perform work effectively". This view of HCI is different from previous work, notably Long & Dowell's "Conception of HCI" (1989) and Dowell & Long's "Conception for HCI" (1989), in that it makes explicit the various potential meanings of the phrase 'general design problem'. This difference, implicitly acknowledged in Long & Dowell (1989), follows from a less extreme position with respect to assumptions about the determinism of HCI's design concerns. Long & Dowell state that the discipline of HCI addresses the '*general design problem* of humans interacting with computers to perform work effectively" p. 15 (1989). The 'general design problem', here, implies a set of public, shared problems which are of interest to the HCI community as a whole, regardless of which form of the discipline individuals perceive themselves as practising (p.18). In contrast, Dowell & Long provide concepts for formulating "a more formal expression of the [HCI] *general design problem* [this author's italics] which an Engineering discipline would address" p.1521. This latter 'General Design Problem' (this author's capitals) is that addressed by only some HCI practitioners, specifically, HCI engineers, and, by implication, not other kinds of HCI practitioner. Indeed, some HCI practitioners explicitly conceive the problems they address in a *different* manner to the 'General Design Problem' (see, for example, Storrs, 1989). In the characterisation of the discipline adopted for this thesis, HCI as a discipline is taken to address 'design problems' to make clear the distinction between the 'general design problem' (community-wide set) and The 'General Design Problem' (super-ordinate class for engineers)[3], and yet explicitly accommodating for both.

1.1.3. The Precise, Technical View of Carry Forward

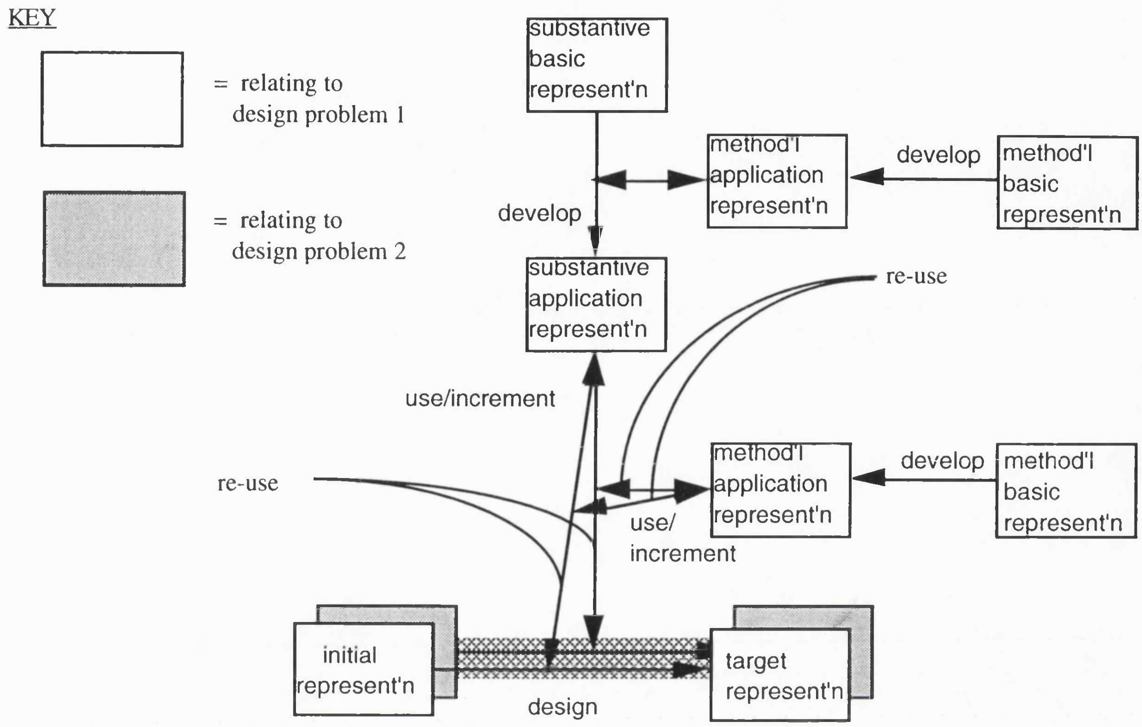
Having initially indicated the nature of carry forward and considered the close relationship between it and the status of HCI as a discipline, let us now consider the nature of 'carry forward' more precisely.

Carry forward, for this thesis, is taken to refer to the development, re-use and incrementation of application representations (see Figure 1.2). To

[3]The distinction between the general design problem and The General Design Problem and will again be considered further later. A clear view of HCI is essential to specifying the kind of carry forward that this thesis seeks, so it is necessary to reveal the interpretation of Long and Dowell's work now. Unfortunately, the use of capitals in this thesis may not be completely consistent with Dowell & Long's use.

comprehend this characterisation, it is necessary to partition the knowledge that comprises the discipline of HCI. HCI knowledge may concern issues of substance, that is, the interaction to be designed and the work to be performed, or methodological issues, that is, *how* the designing is to be done (Long & Dowell, 1989). Knowledge representations may fulfil different functions in carry forward (Long, 1987). A *basic* knowledge representation is one that does not directly support design. It is not referred to by a designer, unless the designer's task is performed incorrectly, or the designer questions the basis for some other knowledge representation which *is* referenced directly. Thus, a basic representation enables and supports the development of other, intermediate, or *application* representations, which *are* directly referenced during design. *Initial* and *target* representations are products of a design cycle, that is, an intermediate step between a client's requirement and the final artefact. The initial representation is the design representation transformed with the support of application representations. The target representation is that produced with the support of application representations. Given the characterisation of basic, application and design representations, 'use' refers to application representations supporting transformation of an initial representation into a target representation. Re-use refers to the use of application representations to support the transformation of, first, one initial representation and, then, another. Incrementation refers to the modification of an application representation in response to its use.

Figure 1.2: Carry Forward as the Development, Re-use and Incrementation of Application Representations to Transform Design Representations



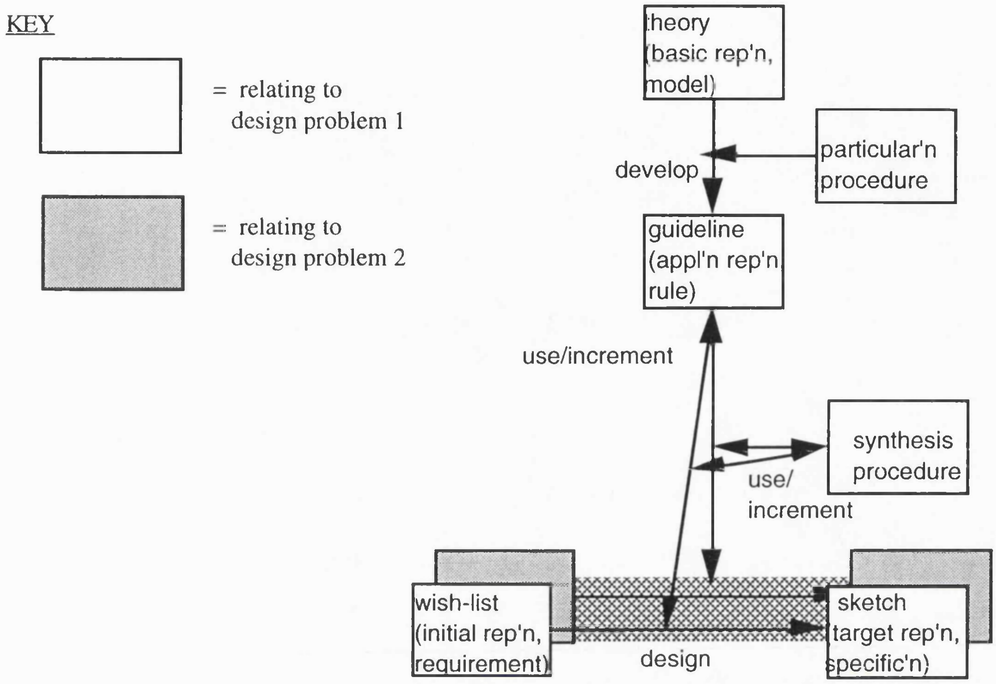
Note that, according to this view, only application representations may be incremented (added to as a result of design) and re-used (referenced in order to design first one instance and then another). Further, a target representation is not incremented or re-used in carry forward, but is enhanced as a result of carry forward. In addition, the development and revision of basic representations are not considered to comprise part of carry forward.

When reporting attempts to realise carry forward (as will occur frequently later in this thesis, it is useful to characterise the form and content of different types of basic, application and target representations. Substantive basic and application representations may take the form and content of frameworks (characterisations of work and interaction), models (representations whose attributes bear systematic relations to attributes of work and interaction), rules (which diagnose or prescribe selected aspects of work and interaction), analyses (which represent problematic features of work and interaction) and specifications (which represent desirable features of work and interaction). Methodological basic and application representations may also take the form and content of frameworks (characterisations of designing), models (representations whose attributes bear systematic relations to attributes of designing), rules (which diagnose or prescribe selected aspects of designing), analyses (which represent the problematic features of designing) and specifications (which represent the desirable features of designing). Initial and target representations may take the form and content of requirements (technical descriptions of a desired artefact) and specifications (technical descriptions of the actual artefact to be, or which has been, implemented).

To illustrate the technical view of carry forward just presented, let us consider carry forward in a manner characteristic of conventional, applied science and as characterised by Long (1987) (see Figure 1.3). In such an approach, a theory (a basic representation, and a sort of model) supports the development of guidelines (application representation, and sort of rule, such as Gardiner & Christie (1987)) by following 'particularisation' procedures. Given a preliminary 'wish-list' (initial representation, and a requirement), the guidelines directly support the production of a design for a prototype (target representation, and a specification) according to 'synthesis' procedures. In this example, the guidelines may re-used - by applying them to the design of other systems. The guidelines may also be incremented, since their use may reveal additional exceptions, application conditions, or novel interpretations[4].

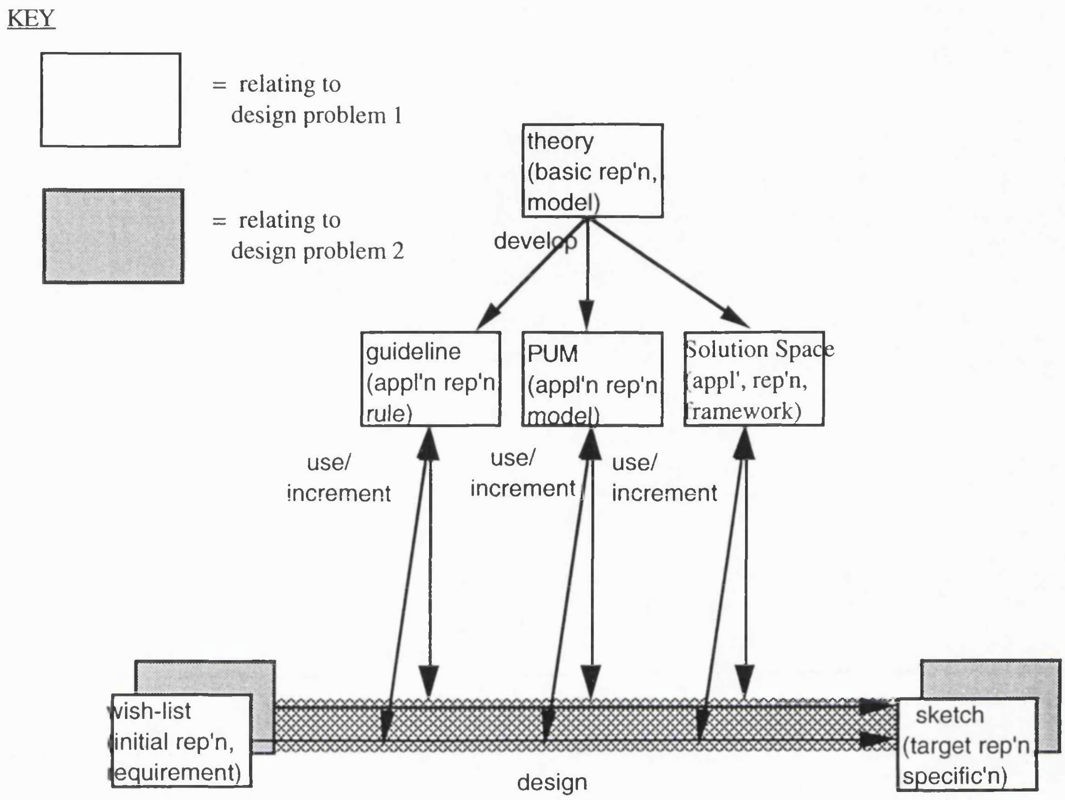
[4]The concern of this thesis is carry forward of *substantive* knowledge. For these purposes, simple reports of carry forward may assist clear exposition. Consequently, future reports of carry forward do not explicitly characterise the methodological knowledge used except to enable the reader to comprehend the report. Such methodological knowledge takes the form of explicit, high level procedures which may be re-used, but not incremented.

Figure 1.3: Example1: Carry Forward in Conventional, Applied Science



A knowledge representation may be used in many ways and involved in carry forward in a number of different ways. It is the task of those seeking to innovate HCI to devise different, technically more advanced application representations, or novel ways of using or incrementing application representations such that the transformation of initial representations into target representations more effectively than previously possible. For example, reasoning with a programmable user model (PUM) (application representation, and a sort of model) may relieve the designer of the difficult task of interpreting guidelines, and so may require less training than reasoning with guidelines (Barnard et al., 1988) (see Figure 1.4). Reasoning with 'Solution Spaces' (application representation, and a sort of framework) (Bellotti, 1993; MacLean et al., 1991) *in conjunction* with guidelines and PUMs may be found to be more effective still.

Figure 1.4: Example2: Carry Forward in Potentially More Effective Applied Science



Opportunities for carry forward are determined by the design cycle. Each transformation between design representations is supported by discipline knowledge and so offers an opportunity for carry forward (see Figure 1.5). The conception of HCI design used in this thesis is a less precise, informal version of Salter's conception (1994). The loosening and informal expression of the conception is required in order to accommodate all forms of HCI. This conception characterises HCI design in terms of non-technical and technical design representations and products. Non-technical products, developed from the user's perspective include a client's requirement and the final artefact. Technical design representations are distinguished as instance requirements, general (class) requirements, general (class) specifications and instance specifications[5]. These design representations and products are linked through processes of interpretation, checking, abstraction, specification, specialisation, instantiation, implementation, obtaining, testing and assessment. There are, then, at most, fourteen opportunities to apply knowledge in support of

[5]General, here, is used in the precise sense of class, that may be specialised, rather than in the looser sense of commonly or widely, that may be used in various ways.

design - two opportunities per transformation, one transformation going forward through the design cycle, one transformation is reverse (see Figure 1.6).

Figure 1.5: The HCI Design Cycle (after Salter, 1994)

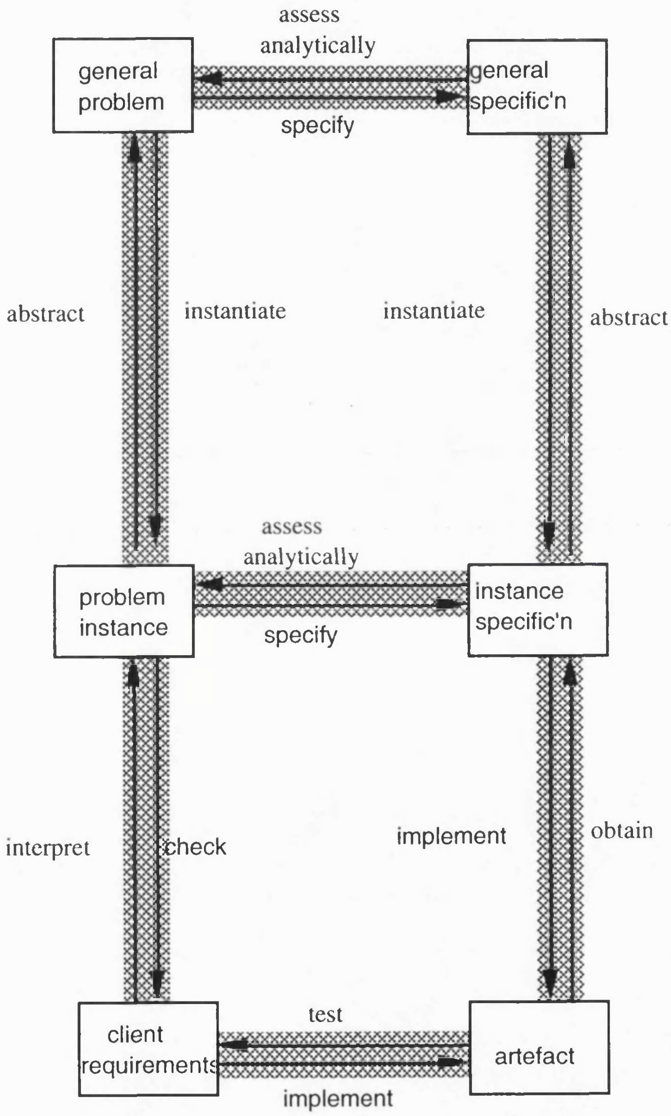
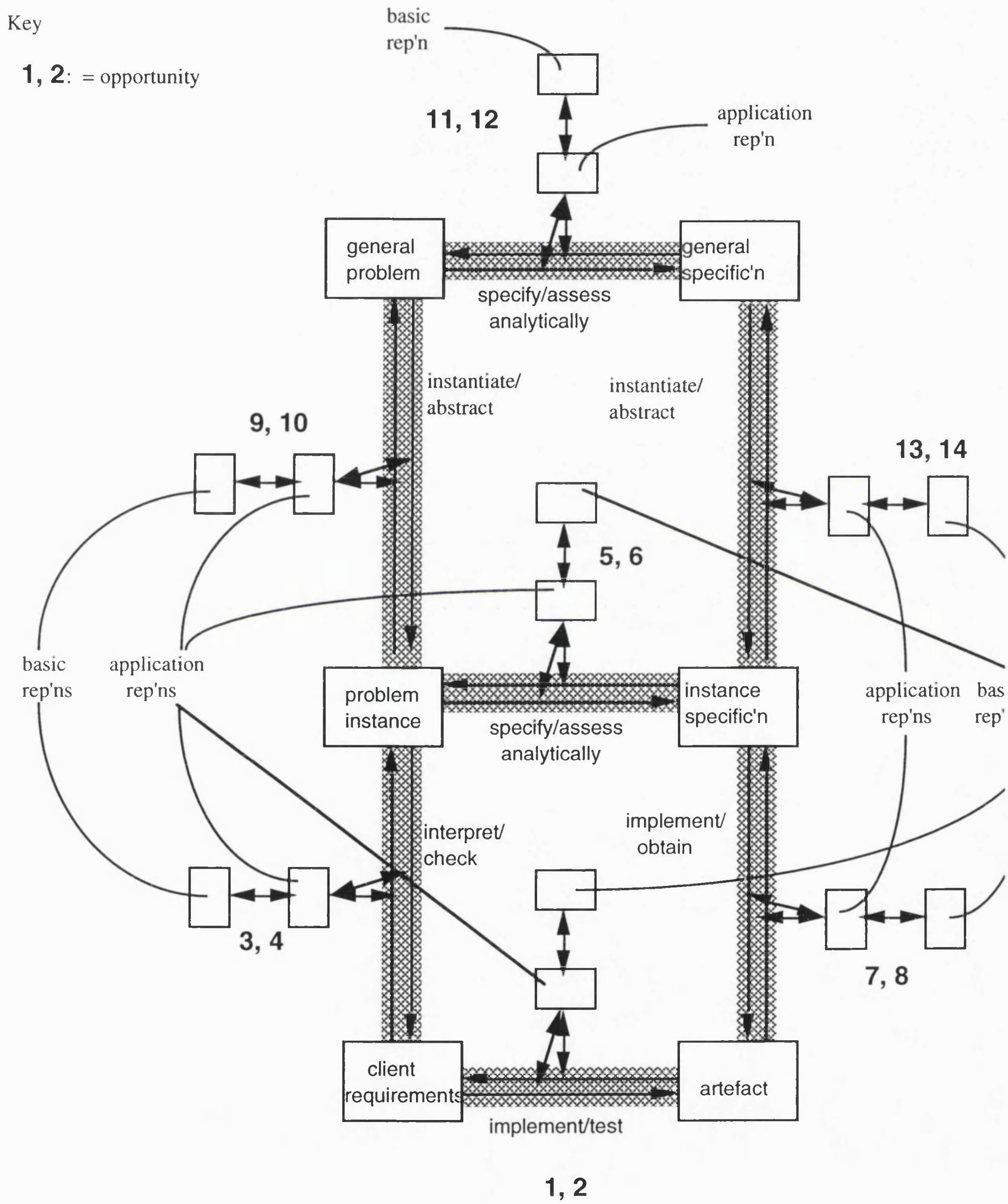


Figure 1.6: Opportunities to Develop, Use and Increment Applications Representations to Transform Design Representations



Given a precise, technical view of carry forward, it is now possible to return to consideration of HCI as a discipline. Further consideration of some alternative forms of HCI, and the associated alternative forms of carry forward, may serve to further explicate difficulties with procurement and suggest the scale and nature of possible responses.

1.1.4. Crafting and Engineering Human-Computer Interactions for C^2

This section characterises HCI both as an immature and a mature Craft discipline, and also as an immature and a mature Engineering discipline[6]. Maturity, here, reflects an assumption that, in practice, some forms of HCI tend to precede, and support the development of, other forms of HCI. Thus, one form of HCI may be particularly associated with, and be regarded as, in practice, an intermediate step to another. For example, in practice, *implicit* knowledge tends to precede, and support the development of, *explicit* knowledge. Similarly, in practice, explicit knowledge of design problem *instances* tends to precede, and support the development of, explicit and informal knowledge of *classes* of Design Problem. Further, *informal* knowledge of *classes* of Design Problem tends to precede, and support the development of, *formal* knowledge of *classes* of Design Problem. Long & Dowell's characterisation of alternative forms of HCI is taken to offer initial characterisations of immature (implicit) craft and mature (formal) Engineering and a starting point for further discipline analysis.

This thesis extends the initial characterisation of immature (implicit) Craft and mature (formal) Engineering, adds characterisations of mature (explicit) Craft and immature (informal) Engineering and pays special attention to carry forward within each form of HCI (see Table 1.1). In essence, the difference between HCI Craft and HCI Engineering Craft is conceived as addressing *instances* of design problem related to other instances in poorly specified ways, whereas Engineering is conceived as addressing *classes* of design problem. Immature craft addresses problem instances *implicitly*, and mature craft addresses problem instances *explicitly*. Immature Engineering addresses classes of Problem *informally*, and mature Engineering addresses classes of Problem *formally*. The informal/formal distinction, here, expresses the fact that knowledge may or may not be acquired and represented according to the rules of verification recognised by the HCI community.

---

[6]Long & Dowell also characterise HCI as an applied science discipline (1989). This thesis does not necessarily disagree with Long & Dowell's analysis, but, for simplicity, does not consider applied science further.

Table 1.1: Characterisation of HCI as Craft and Engineering

	Form of HCI			
Element of Discipline	**Immature (Implicit) Craft**	**Mature (Explicit) Craft**	**Immature (Informal) Engineering**	**Mature (Formal) Engineering**
Knowledge	implicit, fragmented	explicit, structured	explicit, structured, coherent, informal	explicit, structured, coherent, formal
Practices	implementing & testing	interpreting, checking, specifying, assessing, obtaining, implementing & testing	interpreting, checking, abstracting, specialising, specifying, assessing, obtaining, implementing & testing	formalising, verifying, interpreting, checking, abstracting, specialising, specifying, assessing, obtaining, implementing & testing
Design Problems	soft, relations to other design problems unclear	somewhat soft, one of a loose network of design problem instances	somewhat hard, one of a hierarchy of informal General Design Problems	hard, one of a hierarchy of formal General Design Problems
Example Basic Represent'ns	educational background, values, purposes	perspectives, positions	informal frameworks, informal models	formal frameworks, formal models
Example Application Represent'ns	intuitions, experience of extant systems	heuristics, interface styles	informal general principles, informal generic user interface components	formal General Design Principles (following D&L89)
Design Represent'ns	client reqt., artefact	client reqt., artefact, instance reqt., instance spec'n	client reqt., artefact, instance reqt., instance spec'n, general reqt., general spec'n (all informal)	client reqt., artefact, instance reqt., instance spec'n, general reqt., general spec'n (all formal)
Reasoning About Completeness of Target Represent'ns	not open to public scrutiny	with respect to alternative instance representations	with respect to a super-ordinate class representations, but informally	with respect to a super-ordinate class representations, and formally

Thus, informal knowledge is knowledge that is unverified, whereas formal knowledge *is* verified. The correctness of informal knowledge, however, may be assessed in some way, for example, by asking experts' opinions, or working through example problems, but the HCI community would not accept such assessment as verification.

1.1.4.1. Immature (Implicit) Craft

In HCI as implicit craft, interactions between users and their computers are generally hand-crafted, bespoke designs for a single named system produced in response to individual requirements, and often developed through a practitioners' personal, even idiosyncratic, approach to implementation and testing. Such practices are supported by knowledge that remains fragmented, largely implicit and within the heads of 'master' craftspeople (Jones, 1970). The relations between the problem addressed in any instance and other problems is difficult to ascertain, since such relations typically remain implicit. Dowell & Long cite the example of Bornat & Thimbleby developing a text display editor called 'Ded' (1989). The problem instance addressed within implicit HCI Craft is one of the community-wide set of design problems. For example, for Bornat & Thimbleby, the problem was to design a text editor which would enable the user to enter text, review it, add to it, to reorganise its structure and to print it. The problem addressed by Bornat & Thimbleby is self-evidently within the scope of HCI - it concerns the rate at which the computer gives feedback to user key presses, in relation to changes to the position of the cursor or the shape of text characters. Their problem is not an instantiation of the 'General Design Problem' addressed by HCI engineers (super-ordinate class), since the craftsperson's expression of their design problem instance is incommensurate with Dowell & Long's expression of the General Design Problem for HCI Engineering. For example, some concepts from Dowell & Long's 'General Design Problem', such as 'domain' and 'task quality' (see Chapter 2), do not appear in Bornat & Thimbleby's expression of the problem. Bornat & Thimbleby are more concerned with 'interactive behaviours' (which *is* a concept from the General Design Problem) but not to acknowledge any incompleteness or selectivity in terms of Dowell & Long in their problem address. Further, although Bornat & Thimbleby appear to express their problem using concepts that are, on initial inspection at least, compatible with concepts from the General Design Problem, closer examination reveals that apparently similar concepts are, in actual fact, incommensurate. For example, 'ease of use' (Bornat & Thimbleby) appears, on initial inspection, to be similar to 'user costs' (General Design Problem). However, Bornat & Thimbleby's term 'ease of use' presumably *excludes* 'ease of learning' (another widely used concept) but *includes* 'maximum performance achievable by an experienced user' (another widely used concept). Dowell & Long's term 'user costs', in contrast, appears to mean the opposite, that is, to *include* 'costs incurred learning' but to *exclude* the 'task quality' achievable by an experienced user[7].

This thesis also suggests that, in practice, the problems addressed by implicit Craft are 'soft' design problems, that is, problems that are difficult, or impossible to specify well. This addition to Long & Dowell's characterisation of HCI, and its relation to Dowell & Long's conception requires clarification. Dowell & Long suggest that "the extent to which human behaviour is deterministic for the purposes of designing interactive computer-based systems needs to be independently established" (p.1517). They suggest that human behaviour is deterministic in at least some respects and to some degree and usefully so, and so the possibility of an Engineering discipline should not be ruled out, in principle (pp.1519 and 1533). They argue that, if the General Design Problem currently appears to be 'soft', then this may be symptomatic of the early stage of the discipline's development, rather than the General Design Problem's indeterminism and complexity (p1533). This thesis does not disagree with this position. But, it does suggest that, in practice, the softness/hardness of the community-wide set of design problems may vary considerably, due the variable resources available for design (in terms of time, personnel and knowledge). Thus, the super-ordinate class of Design Problems for engineers may not, in practice, be co-extensive with the community-wide set of design problems. Whatever the 'hardness' of the General Design Problem (super-ordinate class) is found to be in principle, so long as design work is constrained by limited resources and rationality is bounded[8], so the General Design Problem and instances thereof will be a sub-set of the range of design problems addressed by the community as a whole. Hopefully, this sub-set will be a large sub-set, but its size is difficult, if not impossible, to determine *a priori*. Indeed, its eventual size may in part depend upon how well HCI Engineering is realised. Thus, this thesis adopts a position that is consistent with the pluralistic sentiments of parts of Long & Dowell, and with the assertions that determinism of design concerns is a

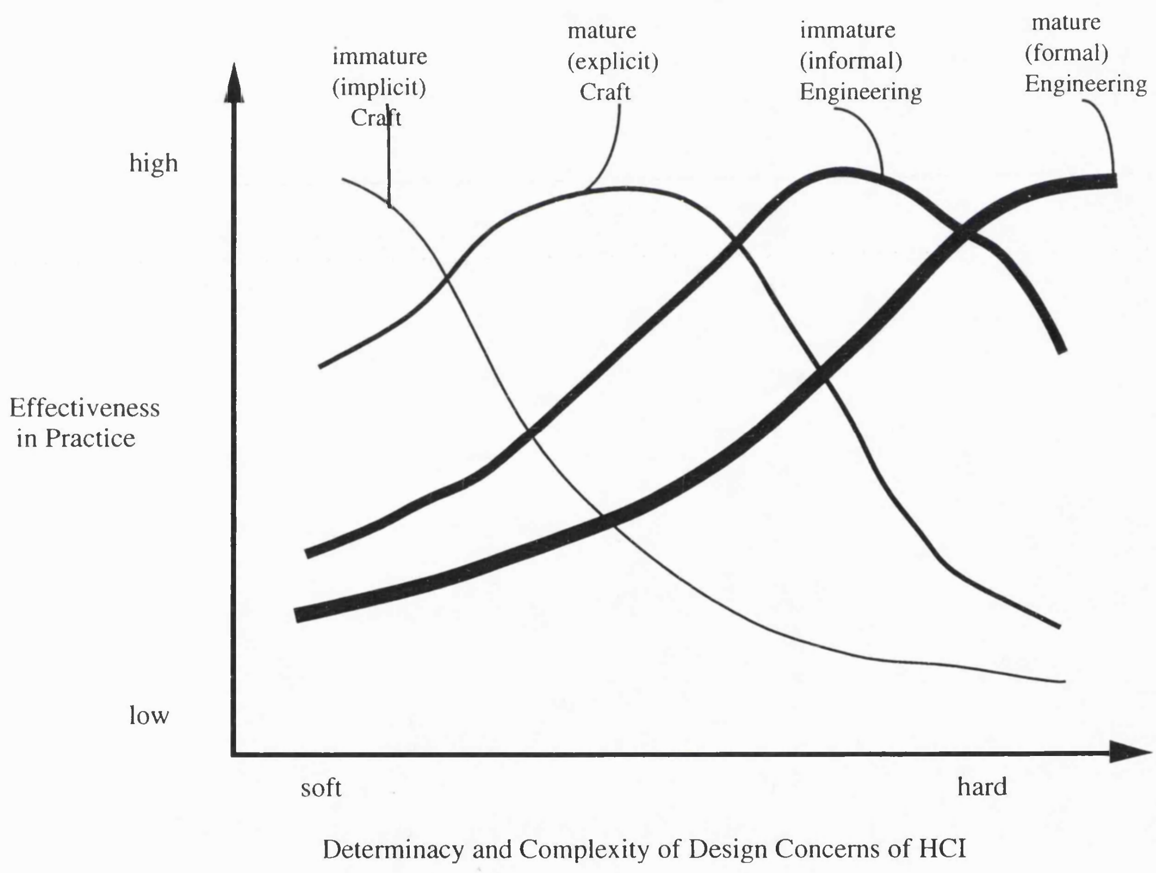
---

[7]Consideration of the status of Bornat and Thimbleby's design problem is of course hampered by the fact that the scope of the design problems addressed by craftspeople may be difficult to delimit precisely, since the nature of this problem remains largely implicit.

[8]individuals may act rationally, but only within the bounds of what they know

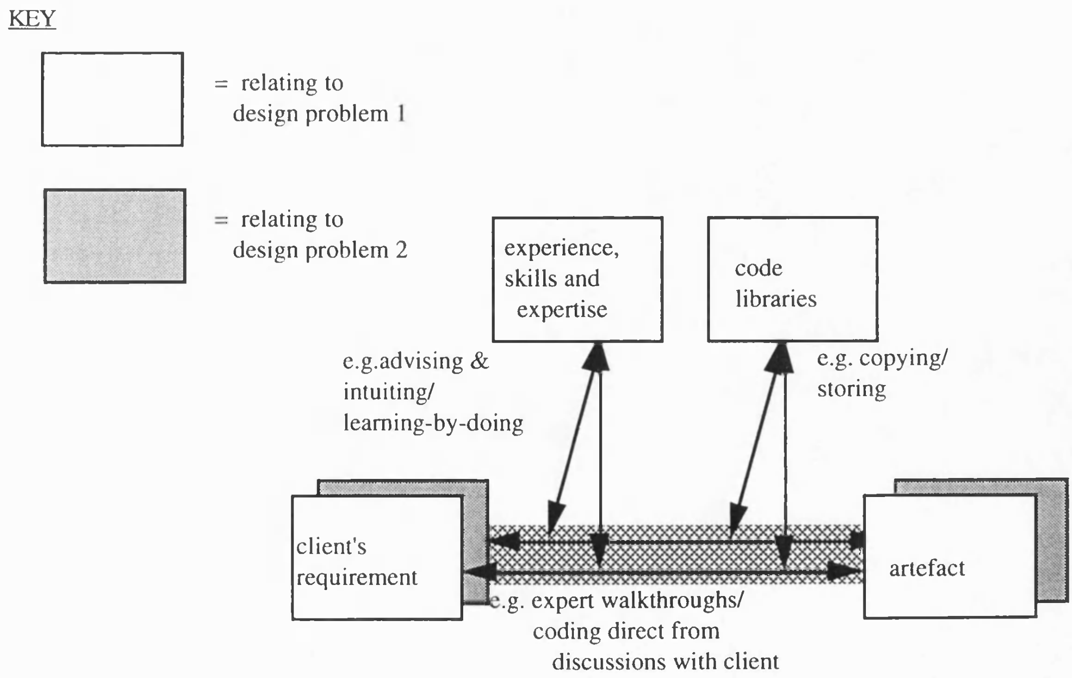
key factor in the effectiveness of alternative forms of HCI and that the actual degree of determinism remains to be resolved. However, this thesis is concerned with effectiveness in practice, rather than principle, and seeks to comprehend all forms of HCI for their mutual benefit (see later), rather than encourage commitment to the development of one form of HCI in particular (formal Engineering) (see Figure 1.7). From the assumption that, in practice, the determinacy and complexity of HCI's design concerns varies, it follows that different forms of HCI are most effective in practice for addressing certain concerns at certain points in time. (This issue is considered further throughout this section.)

Figure 1.7: Differential Effectiveness and Co-existence of Different Forms of HCI



With respect to carry forward in implicit HCI Craft, carry forward may be achieved by, for example, assigning 'expert' individuals to design teams to make their skills and experience available to certain projects via others (Singleton, 1987), storing and copying other artefacts in 'museums', or transferring the knowledge that is implicit in 'best practice' to apprentices during their induction into the HCI community. Such carry forward is 'guaranteed' by the reputation, track record and survival of the human 'experts' that enable it (see Figure 1.8). Long & Dowell give the example of a colleague giving Bornat & Thimbleby some advice (1989). Thus, in implicit Craft, design work (if it may be called such) transforms a client's requirements into an artefact directly, through a process of implementing and testing (prototyping). The application representation supporting this transformation is the designer's own experience and expertise. It is not difficult to perceive how reliance on such mechanisms may fail to achieve carry forward. For example, the interests and movements of human experts may not match project needs, artefacts may be difficult to access or be protected by Intellectual Property Rights, and induction may be incomplete. Thus, lack of carry forward is an inherent risk in immature HCI craft, if not an integral part of this particular form of HCI. Further, lack of carry forward is tied to the fact that, since implicit Craft knowledge is implicit, the knowledge itself is not operational, generalisable or guaranteed to achieve its intended effect (Long & Dowell, 1989). Carry forward relies on intelligent reasoning by designers, which, unfortunately, is not available to public scrutiny and the designers themselves may only have limited insights into their thinking.

Figure 1.8: Carry Forward in Immature (Implicit) HCI Craft



The suggestion that implicit Craft provides only an implicit, highly personal and concrete manner of carry forward does not imply that implicit Craft does not have a place or is to be rejected as necessarily ineffective. On the contrary, as a matter of fact, and for at least some problems some of the time, implicit Craft may be the most effective form of HCI available. Some design problems may turn out to defy specification in principle, or conducive to implicit Craft in practice. For example, it may be a novel problem, posed by an idiosyncratic client who requires an artefact immediately and does not require absolute reliability. Although, in principle, the problem could be better specified and addressed more effectively later - later may be too late for some design work. Thus, C^2 design as implicit Craft may constitute a combination of knowledge, practices and problems that may be effective within certain technical realities and logistic needs. In this sense, HCI as implicit Craft may constitute a necessary and coherent response to such realities and needs. Indeed, implicit Craft appears to have been responsible for many innovative and effective interactions and for the rapid spread of Information Technology in C^2 over recent years. Immature craft may also offer useful support to other forms of HCI (see later). The difficulty is that, since, in practice, implicit knowledge and practice tends to precede explicit knowledge and practice, carry forward through people is too often the only manner of carry forward available. That is, since implicit Craft tends to emerge before other forms of HCI, the only practical way of developing artefacts may be directly from client's requirements and only a single manner of carry forward - through experience and expertise - may be available in support. Implicit craft offers just one of the opportunities to carry forward that, taken together, all the forms of the discipline of HCI could offer (see Figures 1.8 and 1.6). But at least it offers one.

Generally speaking, implicit HCI Craft appears to characterise much of HCI as it has been practised with respect to C^2 systems. C^2 systems are undeniably large, complex and difficult to specify, and thus, without considerable analysis, pose generally 'soft' design problems. C^2 systems also tend to be bespoke systems and thus instances. The need for iterative implementation and testing to ensure the effectiveness of C^2 systems also offers a plausible explanation for some of the difficulties encountered with procurement, particularly re-design during and after installation, and project slip. Lack of carry forward in C^2, then, may arise as a result of practising C^2 design as *only* an implicit Craft. The technical meaning of Jordan, Lee & Cawsey's phrase 'lack of carry forward' (1988) is 'carry forward in only the implicit craft manner'.

At this point, it should be clear that, since lack of carry forward is associated with the early stages of a discipline's development, attempts to achieve greater carry forward in C^2 must seek to progress HCI as a discipline. Greater carry forward requires the construction of combinations of knowledge, practices and problems that are fundamentally different from that of the implicit Craft. There is a need to devise knowledge representations (both basic and applications), ways of designing with knowledge so represented, and design problems that may be effectively addressed by such knowledge and in those ways. Attempts to modify a single element of the discipline in isolation from the others are unlikely to achieve the result required. Indeed, such unco-ordinated interventions may disrupt the coherence of implicit Craft and threaten its effectiveness. For example, making implicit Craft knowledge explicit may have little impact on design in the absence of any new practices to apply this knowledge, or problems to which it may be applied. Any explicit knowledge acquired may fail to be recruited by the unmodified, immature practices of implement and test, or fail to be carried forward by human experts through advice giving and induction into the community. Effort spent making craft knowledge explicit may simply be wasted unless its associated practices and problems are also considered and modified together.

It should also be clear that attempts to progress HCI as a discipline may not assuage all of the difficulties with procurement all of the time. If a more 'advanced' form of HCI is developed, then this more 'advanced' form may be unable to address *all* the C^2 design problems that are currently addressed by the implicit Craft[9]. For technical or logistic reasons, some C^2 problems may be so 'soft' and poorly specified that any approach other than hand-crafting may be frustrated, and so lack of carry forward may have to be accepted and accommodated. However, the same assumptions about the

---

[9]The term 'advanced' here reflects the view that some forms of HCI may, in practice, be pre-requisites for the development of other forms of HCI. It is not suggested that replacing experienced, skilled craftspeople with reasoning with formal principles is necessarily more sophisticated or universally preferable.

determinism of HCI's design concerns also suggest that it is reasonable to expect progress at the discipline level to assuage at least some difficulties with procurement, if not many difficulties. It is necessary to remember that part of progressing HCI is the selection of 'hard', well-specified design problems that may be effectively addressed by more ‘advanced’ knowledge and practices, and the directing of more ‘advanced’ HCI towards these problems. Thus, additional manners of carry forward may be realisable for at least some design problems, if not all.

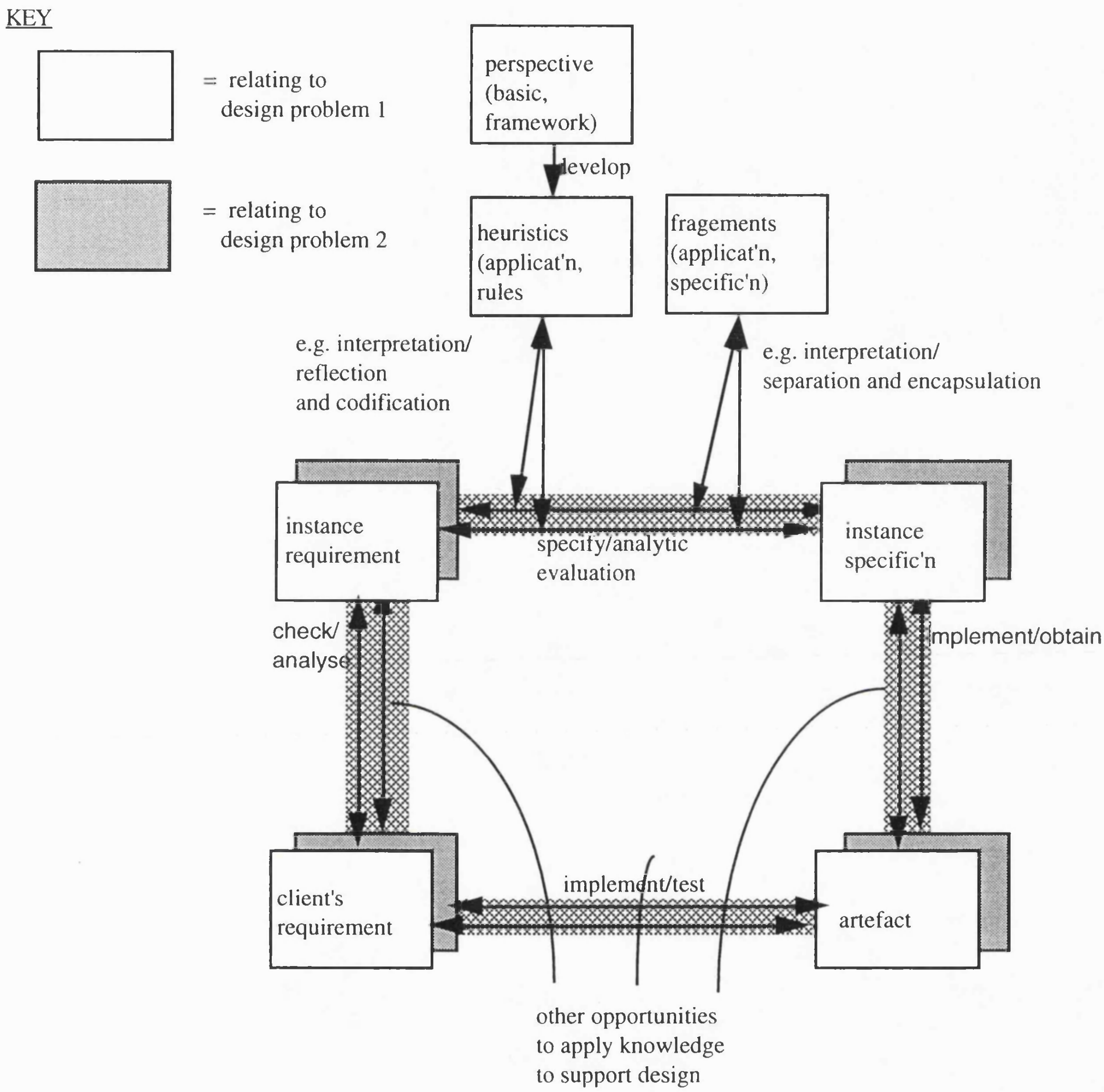
1.1.4.2. Mature (Explicit) Craft

Consideration now turns to the alternative forms of HCI and their means of carry forward. In explicit HCI Craft, knowledge may be characterised as explicit and structured, rather than implicit and fragmented, and as supporting practices of checking, interpreting, obtaining, specifying, implementing and testing, rather than just implementing and testing (see Table 1.1). Further, such practices are public and explicit, and may accord to common or negotiated 'standards'. Importantly, design work involves the production of an explicit instance requirement and explicit instance specification devised in response to the requirement. These representations are distinct from the client’s requirement and the artefact. For example, they may be expressed in a technical language, rather than common English. Like implicit Craft, the design problems addressed are instances drawn from the community-wide set, but they are taken to be somewhat soft, rather than soft, and to be related to other problem instances in at least some respects. Possible relations between problems are complex and varied, depending upon the problems concerns. Manners of carry forward include 'heuristics' (rules of thumb[10], see later this section) and explicit fragments of analyses or specifications of other systems (see Figure 1.9). Heuristics, analyses and specifications may be brought to bear on the problem at hand by interpretation for the purposes.

---

[10]alternatively referred to as 'guidelines', but acquired in a different manner to applied science guidelines (see section 1.1.3).

Figure 1.9: Example of Carry Forward in Mature (Explicit) HCI Craft



Let us consider the example of heuristics in depth[11]. Heuristics are widely believed to be one of the most useful and instructive forms of HCI advice (Smith, 1986). Ways of using heuristics to support walkthrough evaluations of prototypes have also been devised, and these techniques are apparently cost-effective and enable designers to identify 25%-40% of usability errors (Nielsen, 1989). In explicit HCI Craft, carry forward may be 'guaranteed' by seeking to identify the weight of informed opinion in the area. Some elucidation of this point may be required. Craft knowledge in the form of heuristics or rules of thumb, may be made explicit by reflecting upon, and

[11] Readers interested in the example of re-usable fragments of specifications are referred to Sutcliffe, 1993.

then abstracting from, design experience. Such design experience is unlikely to have been acquired under controlled conditions, and thus the identification of systematic, causal relationships between design concerns may not be inferred, at least, not in a manner that may be verified. However, a designer may be able to form an opinion about such relations, and this explicit opinion may be better informed than opinions expressed by individuals with less relevant experience. Although informed opinions do not have the same status as, say, scientific theories, such opinions are nevertheless useful and preferable, in some circumstances, to uninformed opinions or implicit opinions. Further, such opinions may be documented and the opinions associated with different designers and different systems accumulated. As informed opinions accumulate, certain opinions may receive independent confirmation. For example, many authors working on similar, but different C^2 systems in different countries may express similar opinions about reasons for their systems' effectiveness. There is more reason to believe in the correctness of the independently confirmed heuristic than an opinion for which there is no consensus. Thus, although the heuristics in themselves do not guarantee their effects upon design, additional, but unvalidated reasoning by expert 'knowledge users' about the weight of heuristics may provide some reason to believe that the heuristics, if interpreted appropriately, may have the desired effect. Heuristics may also have different scopes (address different ranges of design problems) and may reflect different perspectives on design problems. For example, Colbert's database of heuristics for the development and application of distance learning systems (1993) comprises general heuristics on distance learning systems generally, and medium level heuristics concerning learning systems based upon audio-visual technology, information technology and mixed audio-visual/information technology. The database also includes more detailed heuristics on audio-tele-teaching, audio-visual tele-teaching, text computer conferencing, audio-graphic computer conferencing, computer-based training packages and computer-based collaborative learning systems. Heuristics of each type are grouped according to whether they reflect the purpose of system selection, system design or course delivery. Other authors have devised heuristics for computer systems generally (Brown, 1989) and for specific types of system, such as data-bases (Craig, 1994) or on-line documentation (Horton, 1990).

Explicit HCI Craft may be illustrated with reference to the work of Tetzchner, who analysed, specified, implemented and tested a consistent user interface for a telecommunications environment (1993). A telecommunications environment, according to Tetzchner, is a number of telecommunications services, such as (audio) telephone, video-phone, electronic mail, fax, and video-conferencing, within a single software package. A number of telecommunications services were analysed using a standard notation (CCITT). Further analysis identified and re-described in more general terms, the user inputs and system feedbacks that were common across services. For example, analysis of extant telecommunications services revealed that, when interacting with a conventional telephone, it is usual to 'wait for the tone and dial a number', when sending electronic mail it is usual to 'detect a prompt indicating that the message has been completed and type an alphanumeric code', and when sending a telefax it is conventional to 'wait for the the display to clear and type in a number'. Further analysis described these interactions as 'display 'Connected to Network' feedback and enter address', 'display 'Document ready for sending' feedback and enter address' and 'display 'Document ready for sending' feedback and enter address' respectively. Note that the user action 'enter address' is common to all services and 'display 'Document ready for sending' feedback' is common to e-mail and fax. Identifying common aspects of interaction enabled certain features of the user interface to the telecommunications environment to be re-used, and thus a consistent interface designed. Having analysed a range of telecommunications services in this manner and represented each in CCITT notation, a number of sets of general guidelines for the design of user interfaces were accumulated and reviewed, including Denley et al. (1993), Scapin (1990) and Smith & Mosier (1986). More specific guidelines applicable to selected interactions, such as integration, starting-up applications, displaying status information, resource utilisation and co-ordination, were then developed. An example user interface designed within the MS Windows interface style was then implemented and evaluated. It is notable that the design problems that Tetzchner addresses are relatively explicit and structured, and expressed in terms of standard notations. The problems addressed are also explicitly related to a small number of other problems through their conception as 'telecommunications services'. The notion of 'telecommunications service', however, is barely developed.

"The term system [refers to] the telephone system, telecommunication hardware and software applications, or in other words the telecommunication system as seen from the user. .... The term service is used liberally and can in addition to the normal meaning in telecommunication, that is, a service offered by the telephone company, also mean applications making use of these services."
Von Tetzchner 1993, p. 7

For practical purposes, the notion of 'telecommunications service' is best communicated through the examples given and the reader's experience of systems such as fax, e-mail etc.

In explicit Craft, then, carry forward may evidently be achieved through, for example, heuristics and fragments of specifications. This manner of carry forward is additional, since the means of implicit Craft carry forward (copying and intuiting) may still be applied. As such, the development of heuristics, and re-usable analyses and specifications for C^2 may offer one response to the difficulties of procurement. However, carry forward in explicit Craft is not without its problems (Klein & Bresovic, 1986). For example, interpretation of heuristics is a relatively complex and indirect process, particularly when general knowledge is applied to a specific problem. It was for this reason that Tetzchner accumulated heuristics with many levels of generality. Empirical studies also suggest that designers find guidelines difficult to read, comprehend, and apply (Tetzlaff & Schwartz, 1991; Scapin, 1990; de Souza, Long & Bevan, 1990) and that, because of these difficulties, carry forward may not always occur. For example, in one survey, 80% of designers reported that they believed guidelines would have facilitated the design of a particular interface under consideration, but 63% reported that no guidelines were, in fact, used (Smith and Mosier, 1984). Further, the organisation of 'libraries' of specification fragments is, in practice, extremely difficult and relevant re-usable specifications are notoriously difficult for designers to find (Walton & Maiden, 1993)[12].

The suggestion that carry forward by means of heuristics and re-usable analyses and specifications is difficult does not imply that explicit Craft is

---

[12]These problems are considered further in sections 3.2. and 4.3.

always, in practice, more effective than implicit Craft. As indicated earlier, the relative effectiveness of implicit and explicit Craft depends upon the determinacy of the design concerns addressed, both in principle and in practice. For example, Life et al. developed a method and tool for the empirical assessment of future speech technology because the application and basic representations currently available were perceived to be inadequate (1990). But, if sufficient determinacy is present, then mature craft may offer more opportunities to develop, apply and increment knowledge in support of design (specifically, six more opportunities (see Figure 1.9). To the way of responding to client's problems *implicitly*, a way of responding to client's problems *explicitly* has been added. For any development project, if resources permit, one way of conducting HCI may duplicate another, and so the outcome of each may provide some, independent confirmation of the outcome of the other. If resources do not permit such multiple approaches to design, then at least the project may make a choice between approaches, possibly informed by some appreciation of their relative effectiveness as responses to the client's requirement at hand. From the point of view of discipline progress, a discipline comprising implicit and explicit Craft is more complex and more mature[13] than one which comprises implicit Craft alone. Further, it seems reasonable to suppose that a more complex discipline of HCI is more effective than a simpler one, since both increasing the opportunities to apply knowledge to design and enabling a choice about which of these opportunities to pursue, seems likely to result in more appropriate responses to client's requirements.

Within C^2, there is some movement towards C^2 heuristics and re-usable analyses and specifications. For example, Smith & Mosier's previously cited heuristics were originally developed for the U.S. Dept. of Defense, and specifications for separable user interface objects for C^2 systems have been suggested (Braim & Hepworth, 1992). Thus, there is some evidence that discipline progress has been achieved in at least some instances. It is hoped that such progress will spread far and rapidly (but, of course, not to problems that are so soft that 'advancing' the manner of their address would be counter productive). One coherent response to the difficulties of procurement, it must be presumed, has been already initiated. However,

[13] In the sense that, in practice, explicit Craft tends to be preceded by and supported by implicit Craft

since the remarks about lack of carry forward were made fairly recently (1988), it is reasonable to suggest that even implicit and explicit Craft together fail to meet the perceived need.

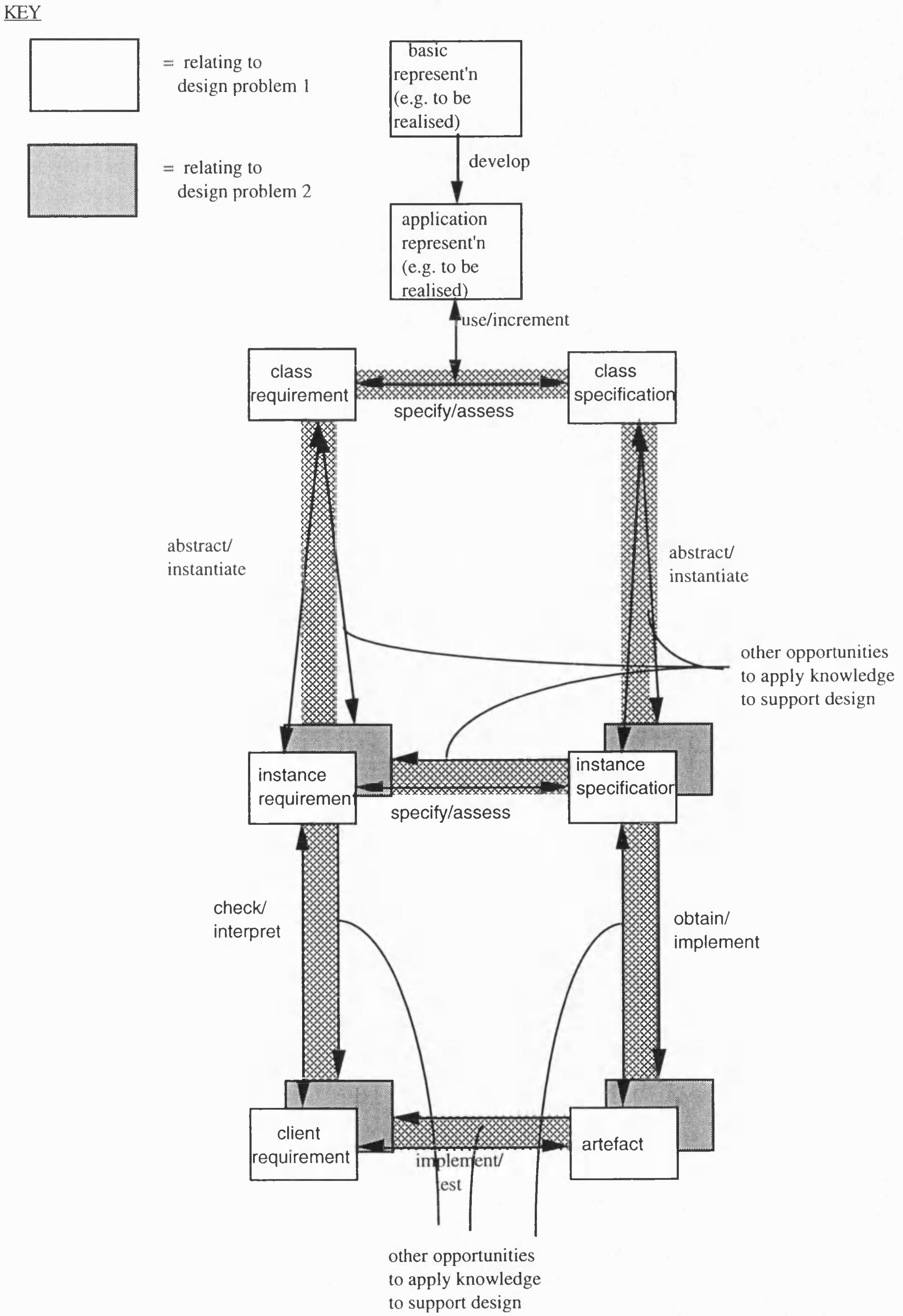
It may be appropriate, then, for this thesis to seek to continue such progress and to develop a form of HCI that is more 'advanced' than HCI as explicit Craft.

1.1.4.3. Immature (Informal) Engineering

Potentially, human-computer interactions may be developed by a form of HCI that is more 'advanced' than explicit Craft - informal HCI Engineering. Informal HCI Engineering may be characterised as explicit, structured but informal knowledge supporting informal interpreting, abstracting, specifying, instantiating, obtaining, assessing, implementing and testing. Design work involves the production of informal general (class) requirements and informal general (class) specifications, in addition to client requirements, instance requirements, instance specifications and artefacts. Informal Engineering addresses classes of design Problem, and instances of classes of problem. However, informal Engineering may only address design Problems than are somewhat hard - softer problems defy formulation as instances of more general classes (see Figure 1.10).

The advantage of informal Engineering with respect to explicit Craft is that, informal Engineering, provides an *additional* means of reasoning about the completeness of design representations in any instance - completeness in any instance may be reasoned about with respect to the relevant general representation. In explicit Craft, one may reason that a previous instance requirement or instance specification concerned issues 'x, y and z' and was not obviously incomplete, so the products relating to a subsequent problem instance (which is presumed to be like the previous case in some unspecified ways) should also concern issues 'x, y and z'. However, since the relations between initial and subsequent requirements and specifications are not fully explicit or coherent, it is difficult to assess how complete the subsequent development products will be. In the absence of any explicitly declared constraints, it is also difficult, or impossible, to rectify any inadequate perception of the similarities and differences between the previous and forth-coming evaluations.

Figure 1.10: Carry Forward in Immature (Informal) HCI Engineering
(Examples to be Realised)



In contrast, in informal Engineering, if the Problem addressed is an instance of an explicitly conceptualised class, then claims about completeness may be reasoned about with reference to the class. For example, if the class design representation indicates that issues 'a, b and c' are of concern, then, any instance requirement or instance specification that was claimed to be complete would be expected to concern instantiations of 'a, b and c'. The class approach fails if the class development products are inadequate - if the class is conceived incompletely, then the instance will be also. Of course, reasoning about the completeness of instances with reference to classes does not prohibit reasoning about completeness with reference to previous instances. If resources permit, one approach to reasoning about completeness may duplicate and confirm or disconfirm another. If resources do not permit, then an informed choice may be made as to which approach best responds to the client's requirement.

The advantage of informal HCI Engineering with respect to carry forward is that yet more opportunities to apply knowledge to design have been provided. These opportunities are opportunities to support abstraction, instantiation and specification and assessment at a general level. Another route from client problem to artefact is available.

The difficulty with informal HCI Engineering is that basic, application and design representations that concern classes of design Problem and instances of design Problem have yet to be developed and put to use. Consequently, and currently, the additional means of reasoning about the completeness of instances and the additional opportunities for carry forward remain speculative. Informal HCI Engineering is not a current reality, but an interesting possibility.

The realisation of this possibility is the objective of this thesis. Realising carry forward in an informal Engineering manner, it is considered, would respond to the perceived inadequacies of carry forward in an implicit or explicit Craft manner. However, since it is not assumed that all HCI design concerns may be addressed by informal Engineering, part of working towards informal Engineering is the identification of problems that may be sufficiently hardened to enable effective address.

1.1.4.4. Mature (Formal) Engineering

To complete this consideration of alternative forms of HCI, let us consider finally, and briefly, mature (formal) Engineering, the form of HCI Engineering originally envisaged by Dowell & Long and that they argued to be, in principle, the most effective. Mature engineering, here, comprises knowledge that is explicit, structured and formal. As a putative example formal Engineering knowledge, Dowell & Long suggest that of Formal Design Principles, which, if developed, would express formal relations between alternative system behaviours and performance. Such knowledge would be generalised (so their application would be direct and efficient) operationalised (so their application would be specifiable) and tested (so their application would be of guaranteed efficacy).

The difficulty with mature formal Engineering is that it may only address hard design problems and that, currently, it remains to be realised. Given the assumption of this thesis that informal knowledge will, in practice, tend to precede and support the development of formal knowledge, and its goal of realising informal Engineering, this thesis may be regarded as an intermediate step towards formal Engineering. From this thesis' point of view, attempts to realise formal Engineering directly from the current state of the Art are somewhat ambitious.

1.2. Zero-->Building a Manner of Carry Forward: Informal Engineering Now!

Given the alternative forms of the HCI discipline, the problem of carry forward, and the state of the Art (mostly implicit Craft with some explicit Craft), this thesis seeks greater carry forward through the development of a form of informal HCI Engineering. Informal Engineering represents a fundamental 'advance' over current means of carry forward. The difficulty is to achieve this objective with the limited resources of a PhD.

1.2.1. Strategy: Balanced Modelling and Practising with Evaluation

The strategy for the development of informal HCI Engineering is 'Zero-->build'. That is, an attempt will be made to develop, increment and re-use application representations that concern classes, and instances of classes of design Problem to support the abstraction and instantiation of general requirements and specifications <u>immediately</u>. The strategy is referred to as

Zero-->build, because the starting point for this thesis is taken to be an operational difficulty (carry forward in C^2 system design), HCI practised as a Craft, a high-level characterisation of HCI as a discipline (Long & Dowell, 1989) and a set of concepts for formulating mature Engineering Design Problems (Dowell & Long, 1989), and the output of this thesis are reports of the development incrementation and re-use of application representations of the desired kind and in the desired way. In the sense that this thesis seeks to realise a manner of carry forward of which there was none before the thesis began, informal HCI Engineering is 'built from nothing'[14].

The Zero-->build strategy is adopted to ensure that the basic representations generated (models of C^2) indeed support the development, incrementation and re-use of application representations in support of the abstraction and instantiation of general requirements and specifications. The strategy was also adopted to ensure that the problems selected (aspects of C^2) are sufficiently hard to be effectively addressed by the form of HCI under development. In essence, Zero-->build provides an opportunity to consider immediately, and on the basis of case evidence, the potential effectiveness in practice of the means of carry forward envisaged. Importantly, such considerations are not simply deferred (cf. Pietras, 1994), or reduced to consideration of effectiveness in principle. Zero-->build ensures that the work proceeds breadth-first and realises preliminary knowledge, problems and practices in combination. Given the inadequacy of current understanding of the relations between HCI knowledge, practices and problems, an attempt to develop one of these elements in depth, but in isolation from the others, is fraught with difficulty. For example, the nature of such knowledge-practice-problem relations are uncertain, and there is some doubt about their very existence. For example:

"At the very least, it should be perfectly clear that highly important theoretical and/or practical uses of a [cognitive] model must be outlined

---

14 Given that an additional output of this thesis is the refinement of the form of HCI to be realised, this thesis may be said to be building from less than nothing. However, in actual fact, much previous work is of relevance, notably: (i) literature about the domain of C^2 (see sections 1.2.1. and 2.1); (ii) assessments of this literature (see section 1.2.1.); (iii) reports of current carry forward and design practice (see sections 3.2. and 4.2.); (iv) assessments of the effectiveness of current carry forward and design practice (see sections 3.3 and 4.4.); (v) characterisations of different forms of HCI (see section 1.1.); and (vi) general material relevant to research strategy (see section 1.1.6 and 1.2).

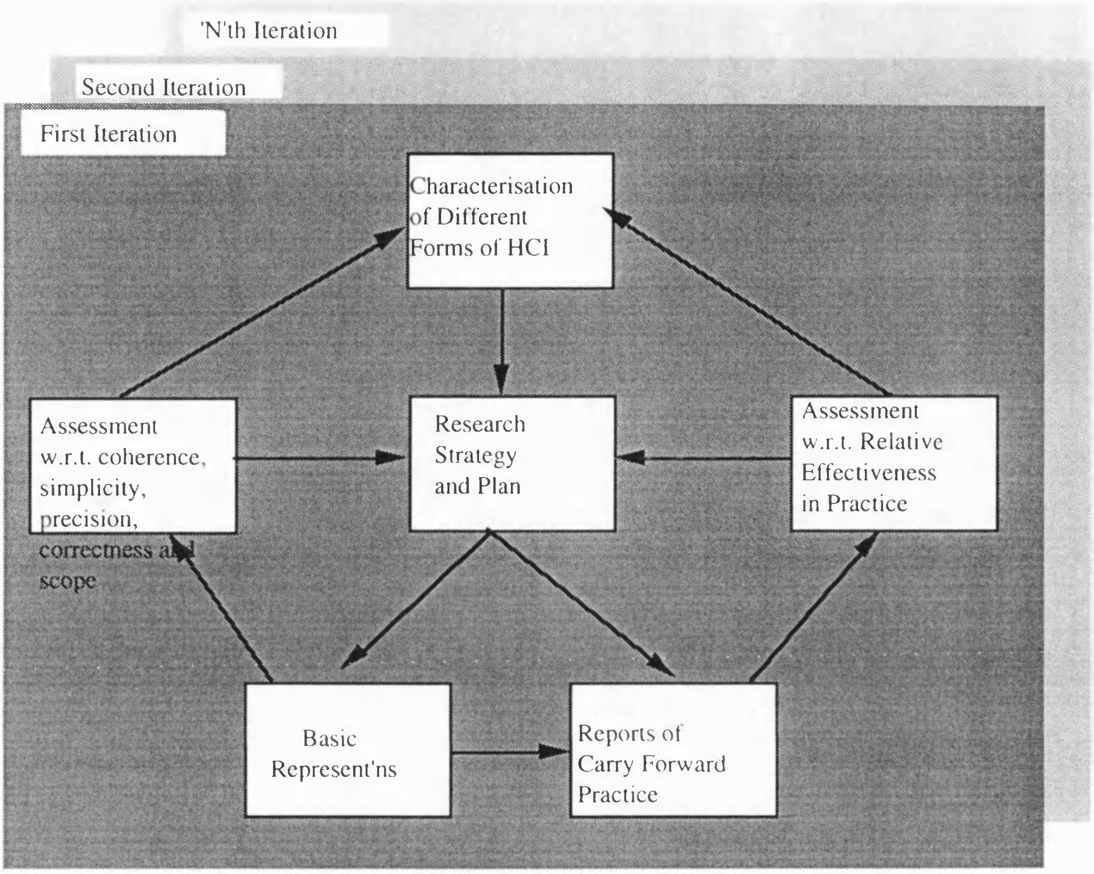
beforehand to convince oneself and others that the apparently risky process of developing a valid, understandable model is worth the effort."
van der Schaaf, 1993, p. 1439

Further, because the design concerns of HCI include highly intelligent human agents, some positions within HCI, such as Systems Thinking, suggests that models of the kind envisaged may only be developed at the expense of their scope, accuracy or fitness-for-purpose (Checkland, 1981). As Dowell & Long say, the determinacy of HCI's design concerns (and so the 'hardness' with which Design Problems may be formulated) remain to be established. In short, there is very little reason to presume that any informal Engineering knowledge acquired will *necessarily* support informal Engineering design practices, or that problems of the appropriate sort are being addressed, or even exist within C^2. Indeed, in the face of relative ignorance of knowledge-practice-problem relations and well-argued skepticism, there is every reason to seek to demonstrate relations between Engineering knowledge, practices and problems, and the effectiveness in practice of the combination.

The Zero-->build strategy was also adopted to provide a contrast set from which potential objectives could be extracted. That is, initial 'builds' (combinations of knowledge, practices and problems) may be compared and contrasted with current explicit Craft alternative, so revealing more precisely characteristics of the more 'advanced' form of HCI desired.

Four streams of work are taken to be necessary elements of any strategy for realising a form of informal HCI Engineering and assessing its effectiveness in practice: (i) Generate Basic Representations; (ii) Assess Basic Representations; (iii) Conduct and Report Carry Forward Practice; and (iv) Assess Carry Forward Practice Reports (see Figure 1.11). An additional workstream is required to refine characterisations of alternative forms of HCI by reflecting upon the outcome of the other workstreams (basic representations, assessments of the coherence, simplicity, precision, correctness and scope, reports of the carry forward practice, and assessment of the effectiveness in practice of the carry forward conducted) and to plan the research work. This strategy is essentially iterative boot-strapping. Zero-->build encourages rapid, rather than slow iteration.

Figure 1.11: A Strategy for the Development of HCI as Informal Engineering: Zero-->Building



The 'Generate Basic Representations' workstream develops the knowledge necessary to enable the development of application representations to be involved in carry forward. The 'Assess Basic Representations' workstream assesses how well the basic representation satisfies the criteria of explicitness, coherence, simplicity, accuracy and scope (Dilworth, 1981). Explicitness, here, refers to the extent to which a basic representation is stated, laid open for scrutiny and made available for dissemination. Coherence refers to the extent to which a basic representation is represented according to the semantic and syntactic rules of the language in which it is expressed. Simplicity contrasts with complexity, and refers to the structural characteristics of the basic representation and its consequences for ease or difficulty of comprehension. For example, a blackboard model of a user comprising two levels of the blackboard and controlled by an agenda would be simpler than a model comprising five levels and controlled by a second blackboard. These criteria concern the expression of knowledge in and of itself, regardless of its origin or purpose.

They reflect the need for any translation to re-represent knowledge without error. Correctness refers to 'validity', the extent to which a model correctly represents the 'real world'. For example, accuracy may concern the extent to which a user possesses knowledge of the kind asserted by a blackboard model of the user. Precision may concern, for a given level of description, the extent to which all different types of knowledge have been discriminated in the model and the similarities and differences explicated. Scope refers to the range of problems addressed by a research output. For example, a blackboard model may be applicable to all users of any computer-aided design system, or just to experienced civil engineering designers using the simulation facilities of mouse-driven computer-aided design systems.

The 'Conduct and Report Carry Forward Practice' workstream develops one or more application representations from the basic representation and uses the application representation to support the design of first, one artefact and then another, and finally increments the application representation in the light of its use. That is, an attempt is made to realise a discipline of a certain sort. The 'Assess Carry Forward Practice Reports' workstream assesses how well the combination of basic, application and design representations in action satisfies the criteria of work effectiveness and design practice efficiency. Work effectiveness refers to the effectiveness with which interactive systems developed perform work. For example, the research that supported the development of WIMP (windows, icons, menus and pointing) (Baecker & Buxton, 1987) interfaces achieved high levels of work effectiveness. Such interfaces greatly eased the management of personal information, compared to say, command line interfaces. Design practice efficiency refers to the costs incurred during development by designers, their equipment and other forms of support. Such costs include the mental effort required, the capital costs of equipment and its maintenance, and the costs of travelling to meet users. For example, the research output 'The User Skills Task Match Method for Requirements Capture' (USTM) would be regarded as more efficient than the current practice of its host organisation, since the method improves team dynamics and communication (Fowler et al., 1989). Taken together, work effectiveness and design practice efficiency express the most salient technical 'gulfs' that may arise between 'research' and 'design' communities within HCI (Buckingham Shum & Hammond, 1994).

The final output of the 'Generate Basic Representation' workstream is a well-formed Engineering model which achieves high levels of explicitness, coherence, simplicity, precision and validity, is well scoped, and supports a wide range of effective informal Engineering practices. Intermediate outputs of modelling satisfy the criteria of explicitness, coherence etc. less well, and support fewer, and less effective informal Engineering practices. As the 'Generate Basic Representation' workstream progresses, it is reasonable to expect practising to become more effective. However, given uncertainty about knowledge-practice-problem relations, any such increases in effectiveness remain to be established.

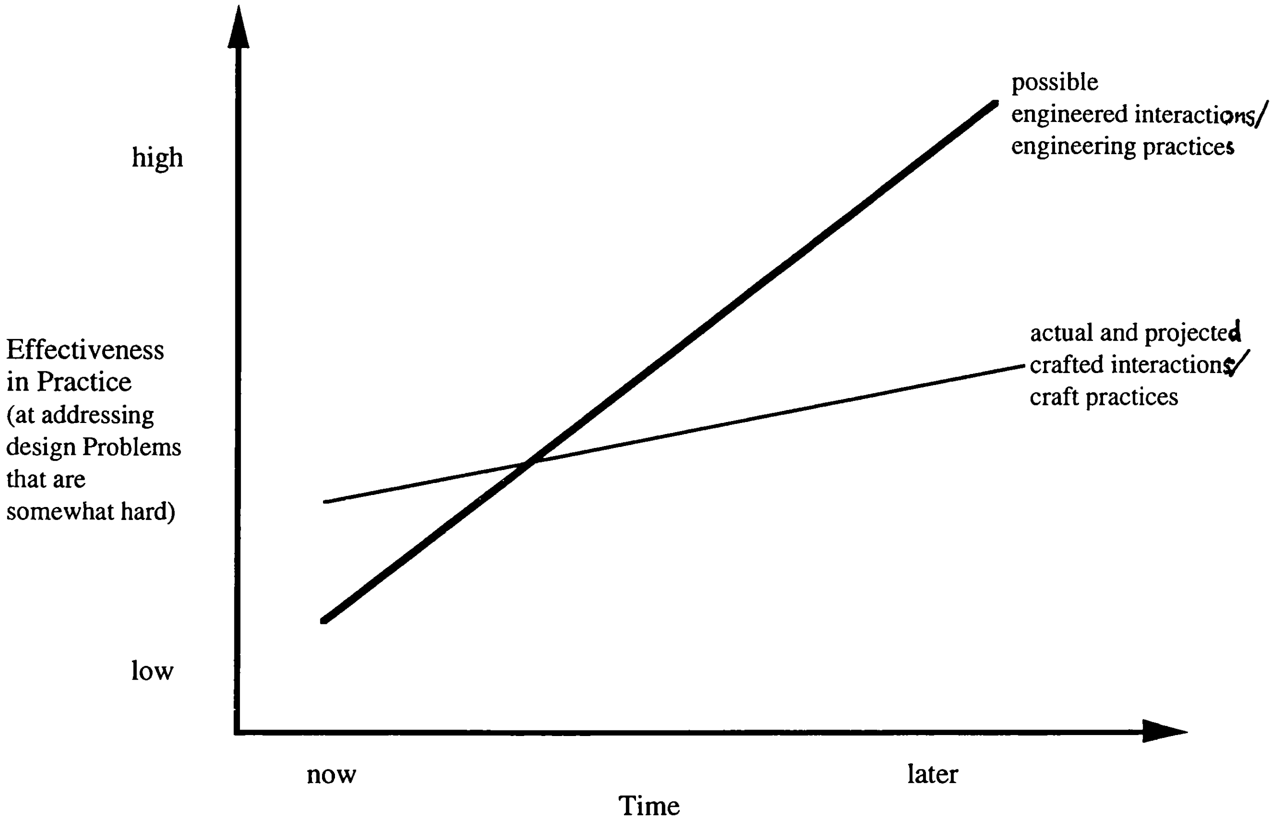
The need to Zero-->build (realise a form of informal HCI Engineering immediately) requires 'Generation of Basic Representation', 'Assess Basic Representation', 'Conduct and Report Carry Forward Practice' and 'Assess Carry Forward Practice Reports' workstreams to proceed in a balanced, 'breadth first' fashion. That is, it is necessary to generate basic representations, and then develop, increment and re-use application representations in parallel, and to approximately equal extents. Given limited resources, even a somewhat unbalanced, depth-first strategy currently risks failing to construct an effective combination of knowledge, practices and problems, and may only realise a selection of these elements. That is, a depth-first strategy may generate a basic representation without indicating the sort of application representations that would be appropriate, or devising a way of using these application representations. The realisation of immature HCI engineering would be delayed, the strategy of Zero-->build jeopardised, and the crucial questions about basic representations and what may be done with them. For example, if modelling was pursued at length and in the absence of practising carry forward, one would not know immediately what kind of carry forward practices the 'mature' model supported, how well it supported them, the type of design representations effectively transformed by these practices.

The need to proceed in a balanced fashion imposes several constraints on the initial output of each workstream. First, the generation of basic representations must focus on the most informative and easily conceived aspects of C^2 and model C^2 in a preliminary fashion. (A rapidly-produced output that focussed upon uninformative aspects of C2 is likely to be trivial). Consequently, any practice supported is likely to be ineffective, and so difficult to present as progress. The most informative and easily conceived aspects of C^2, it is suggested, are the domain objects (see section 1.2.2). The preliminary basic representation is, for this thesis, a preliminary conception. Thirdly, generating a basic representation and conducting and reporting carry forward practice must take priority over their assessment, since the objective of the work is to realise a form of HCI, and in particular, a form of carry forward. The effectiveness of the means of carry forward realised is a crucial issue, but this issue may only be addressed once modelling and practising have reached a certain, minimum stage of development. Fourthly, assessing carry forward practice must take priority over the assessment of the basic representation, since, given the objective of this thesis, the key issue is the nature of the carry forward that a basic representation enables. Although the intrinsic quality of the basic representation generated is a crucial issue, it is only worth addressing once the utility of such a representation has been established. In essence, placing these constraints on the thesis may help to ensure that any attempt to realise informal HCI Engineering is as convincing as possible, given the limited resources.

This thesis, then, is preliminary and selective in nature. However, it is intentionally so, in order to achieve the broad objectives it has set itself within the resource constraints of a PhD. Despite this preliminary and selective nature, such a thesis may nevertheless attempt to construct a combination the principal elements of a discipline - knowledge, practices and problems - and consider its effectiveness.

There is little hope that 'Zero-->build' will realise the most effective form of carry forward in informal HCI Engineering at the first attempt. Indeed, as later chapters reveal, the first attempt fails, and only at the second attempt is anything like the desired practice realised. Whatever the success of effectiveness of the attempts, however, the reports of the attempts will constitute at least initial examples that subsequent work may progress and execute in a more satisfactory manner (see Figure 1.12). Each iteration in the evolution of carry forward practice may make good at least one of the deficiencies of its previous execution, and seek to become continually more precise about the nature of this form of HCI.

Figure 1.12: Crafting and Informally Engineering
Human-Computer Interactions: Progressive Evolution



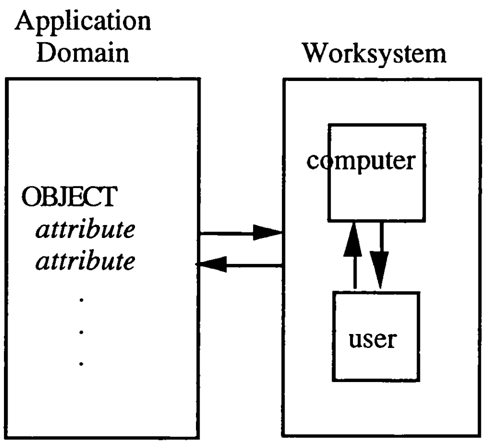
Having specified the objectives and strategy for this thesis, the outputs and methods of the principal workstreams may now be considered in more detail.

1.2.2. Generating a Basic Representation: Abstracting a Preliminary Conception of the Domain of C^2 from the Literature

Thus far, the early output of the modelling workstream has been termed 'a conception of C^2' and characterised as a precursor to a full model. To implement successfully the zero-->build strategy, this output must be informative, easily acquired and preliminary.

The conception seeks to ensure that it is informative by focussing on the *domain* of C^2. To avoid misunderstandings, a domain of application, here, is where work originates, is performed and has its consequences (see Figure 1.14.)(Dowell & Long, 1989). Requirements for change in the state of the domain are expressed through task goals. Task goals are allocated by organisations to worksystems and may be expressed in terms of objects and their attributes. A worksystem comprises a set of interactive user-computer behaviours, which are exhibited in order to influence the state of the domain (perform tasks) and which are supported by mutually exclusive user and computer structures. Effectiveness is expressed as performance, that is, as task quality (how well task goals have been achieved) for costs incurred by the user and the computer. Thus, the domain of C^2 comprises the goals of military work expressed in terms of objects and attributes.

Figure 1.13. A Conception for Human-Computer Interaction
(after Dowell & Long, 1989)



A pre-conception of the domain of C^2 is taken to be informative for the following reasons. Domain knowledge is thought to provide a framework for the development of worksystem models, and a means of simplifying and delimiting such models (Simon, 1981, Woods & Roth, 1988).

"Adaptivity to an environment is their [artefacts'] raison d'etre"
Simon 1981, p. 27

"We can often predict behaviour from knowledge of the system's goals and its outer environment, with only minimal assumptions about the inner environment."
Simon 1981, p. 11

"A man, viewed as a behaving system, is quite simple. The apparent complexity of his behaviour over time is largely a reflection of the complexity of the environment in which he finds himself."
Simon 1981, p.65

Domain knowledge also provides a starting point and a source of inspiration for interface design (Vicente, 1990), particularly direct manipulation interfaces (Shneiderman, 1982).

"For a design to be successful [in the sense of extending the capabilities of computers to more people and making them productive and satisfied in their work], a trinity of elements is needed: knowledge of the principles of good user interface design ..... , knowledge of user work and goals (which is provided by field research methods), and management commitment to usability goals and processes."
Wixon, 1992, p. 53

In particular, knowledge of physical and abstract domain features may enable the design of appropriate interface metaphors (Pejtersen, 1991; Carroll, Mack & Kellogg, 1988) and displays that make <u>all</u> properties of complex systems 'visible' to the user (Rasmussen & Vicente, 1989).

"There are certain cases where it is ... appropriate to begin by investigating the constraints imposed by the work domain".
Vicente 1990, p. 493

"An interface should be designed so as to represent the abstract properties of the process explicitly."
Rasmussen & Vicente 1989, p. 525-6[15]

The C^2 (end-user) community appears to share this belief in the primacy and importance of domain knowledge. 'Domain-oriented' C^2 systems are thought to be particularly effective (Van Crefeld, 1985).

"Historically speaking, those armies have been most successful which ... the Germans, following the tradition of Scharnhorst and Moltke, call 'Auftragstaktik', or mission-oriented command systems."
Van Crefeld 1985, p. 270

The conception is also relatively easy to acquire, since there are many documentary sources of domain knowledge, particularly from disciplines

---

[15]The process referred to here is the industrial process - the domain of a process control system.

other than HCI, such as Military History and the Military Art. For example, Military History offers relevant knowledge in the form of narrative accounts of command in war, and comparative, historical analyses (Millet & Williamson, 1988; Van Crefeld, 1985). In addition, military commanders may write autobiographies (Lawrence, 1935) and so-called 'principles of war' have distilled the historical lessons learned (Alger, 1982). User knowledge includes standard procedures for operating equipment, rules of engagement, officer training manuals and text books (Kiely, 1988), and guidelines for compiling tactical pictures, formulating communications and representing plans. Dictionaries of military terms (Joint Chiefs of Staff, 1988; Noel & Beach, 1988), and the intelligence reports, tactical pictures, communications and plans produced by military personnel may also be included in this category. HCI knowledge includes frameworks, and models of C^2, which seek to characterise and represent 'what C^2 systems do' (Sherwood-Jones et al., 1992; Dowle 1989; Conley 1982). Many procurement documents also reflect domain knowledge, particularly future engagement scenarios (Clancey 1987), user-centred views of requirements, and task descriptions (Tainsh 1985). Finally, there exists definitions of C^2, typically focussing on the notion of 'combat management', and views on the nature of C^2 (Tainsh 1982).

Although many documentary sources of C^2 knowledge are available, the comments of some writers from the C^2 community suggest that a conception of the domain of C^2 is sorely missed.

"The Board noted that there is almost no commonly understood vocabulary or conceptual framework for analysing, designing or evaluating command and control systems" ..... "the major difficulties in developing acquiring and deploying command and control system are not primarily technical, but conceptual (What should they do?) or administrative (How do we organise the required resources?)"
US DoD 1978, in Moll 1982

"There is no adequate foundation for a theory of command and control, and hence no guiding principles for system design and evolution"
US Office of the Secretary for Defense 1979, in Wohl 1981

"The absence of commonly understood concepts of command and control system performance and the existence of language barriers among technologists, policy analysts, planners and commanders, all underlie the fact that we lack in DoD any very useful conceptual framework for evaluating or specifying command and control systems."
Moll 1982, p. 23

"command and control lacks a firm theoretical basis"
Hwang, et al. 1982, p. xvi

"there is no theory ... for command and control of complex systems"
Harris & White 1987, p. xi

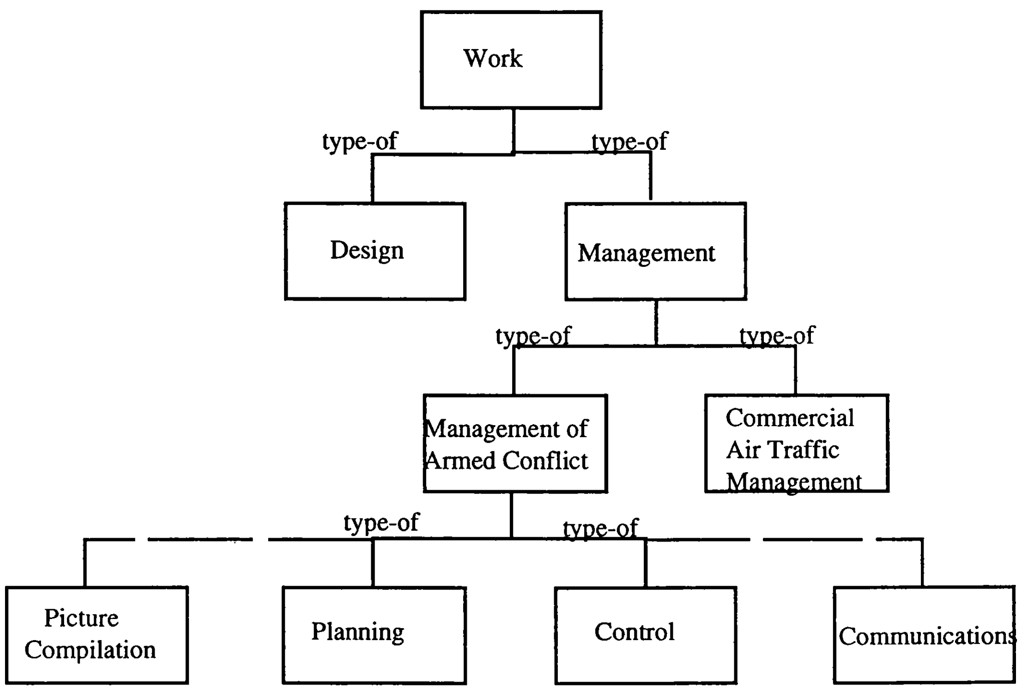
Further, poor design, and consequent operational disasters, have been attributed to lack of an adequate domain conception.

"lessons learned about the 'reasonable choice of disaster' [when a disaster results from deliberate action based on the reasoning that an alternative interpretation is impossible] suggest that theories about the environment lag behind the capabilities we would like to see in our C^2 systems in controlling this environment and that such gaps can lead to disasters"
Lanir, 1991, p. 226
It is clear in the previous paragraph that the distinction between 'historical', 'military' and 'HCI' knowledge of C^2 is somewhat blurred. For example, 'HCI' knowledge may be acquired from military personnel as part of user-centred design, and 'historical' knowledge may be internalised by military personnel when they read principles of war. But the distinction serves to characterise the range of sources available. It is also clear that, with a little assistance from domain experts, informal analysis of documentary sources is likely to yield adequate results relatively quickly.

A conception of the domain of C^2 is also easy to acquire by virtue of its preliminary nature. However, preliminary does not imply slip-shod. A conception is a particular type of basic representation and so one may discriminate between well- and poorly-formed examples. A conception, here, is taken to be a set of concepts for formulating informal design Problems. For example, 'HCI(e)' is a conception for formulating the general Problem of designing humans interacting with computers to perform work effectively (Dowell & Long 1989). HCI(e) presents the concepts through which this problem is expressed, that is, work, the user, the computer, effectiveness etc.

Conceptions may support the formulation of design Problems at many levels of generality, for example, at the level of the class, sub-class and sub-sub-class (see Figure 1.15). For example, a conception of 'management' supports the formulation of design problems concerning managers interacting with management aids to manage operations effectively - a relatively general class of problem. A conception of 'Naval Officers interacting with the 'ADAWS IV' C^2 system to plan and control defence against air-to-surface missile attacks on convoys' supports the formulation of design Problems involving personnel of this specific affiliation and rank performing this specific work with this specific product. It follows that the output of this thesis (a conception of C^2) supports the formulation of a medium level class of design Problem - military commanders and their staff interacting with C^2 equipment to plan and control military operations effectively.

Figure 1.14: Domain Conceptions of Different Levels of Generality



The output of this thesis is, in actual fact, yet more preliminary than a conception in that it is highly selective about the aspects of the C^2 domain it seeks to characterise, hence, reference to it as a 'pre-(liminary)conception'. For this thesis, the pre-conception is selective in that it only concerns one of the principal elements of a design Problem - the domain. Further, the pre-conception is narrow and concerns abstract domain characteristics only. Narrowness, here, indicates that the pre-conception seeks to be general to only some military operations, namely, those which are intuitively central to the notion of military work or those which distinguish military from non-military work. Such central, distinctive operations may be termed 'combat operations', such as anti-surface warfare, anti-air warfare, anti-submarine warfare, amphibious landings etc. The pre-conception does not seek to characterise more peripheral types of military work, such as low intensity operations (counter-insurgency, peace keeping etc.) and operations which are conducted by military institutions but which are closely related to, or more correctly referred to as, civilian work (diplomacy, policing etc.). The focus of the pre-conception on abstract domain characteristics indicates that emergent properties of task goals are of concern, that is, those properties which emerge when considered in the context of the intentions of C^2 systems, rather than their physical properties, for example, whose relevance is less sensitive to a worksystem's intentions.

The pre-conception takes the form of a component (part-of) hierarchy of domain objects, with partial attribute lists for each object. Objects and attributes are briefly elucidated in text. A precise view of the desired form and content of the pre-conception was used as a target for the analysis of documentation and so further increased the rate of development.

1.2.3. Conducting and Reporting Carry Forward Practice: Documenting Illustrative Case Studies

To realise carry forward in a manner characteristic of informal HCI Engineering it is necessary to conduct the practices envisaged. A case study was selected as the form of reporting carry forward practice, since it best supports assessment by comparative review. This means of assessment was chosen in order to comply with the zero-->build strategy. A qualitative, sequential description of the activities undertaken was considered the minimum basis upon which judgements about work effectiveness and design practice efficiency could be formed. The case studies, then, do not seek to prescribe good practice, but rather seek to expose the practice that could be realised on a particular occasion and so enable its assessment with respect to certain criteria. Given that the basis for such an assessment is work that was conducted and reported by an interested party (i.e. the present author), the report focuses upon the products of the work, about which readers may form their own opinion.

The case study reports seek to illustrate particular carry forward practices that are characteristic of a particular form of HCI. Thus, the case study reports only relate the development, re-use and incrementation of application representations to support the transformation of design representations in detail. Activities that enable, follow or in any way fail to comprise carry forward, are reported to the extent necessary for the reader to comprehend the report. Such activities include the development of application representations from the pre-conception and other basic representations and the completion of initial design work. Further, the case study reports only relate to the manner of carry forward selected for illustration. Alternative ways of developing, using or incrementing application representations are not reported, and are not deemed to support the comprehension of the selected practice. The detailed part of the case study reports also only include the aspects of carry forward that implicate the status of the practice as informal Engineering rather than explicit Craft (transformations involving general (class) requirements and general (class) specifications)[16], and the claims made about informal Engineering (additional opportunities to apply knowledge in support of abstraction, instantiation and specification/assessment at a general (class) level. Since the thesis focuses on the carry forward of substantive knowledge, the methodological knowledge applied is not reported except for comprehension, and then only as simple procedures.

In this thesis, the case studies report carry forward practice as conducted by the author. That is, the author personally trialled the pre-conception to see how well it enabled the development of certain application representations and how easy these application representations were to use

[16]The status of the reported practice as immature Craft or formal Engineering is not considered, since it is taken to be self-evident that the work reported is explicit and informal.

and increment. An alternative approach would be to randomly assign designers to different groups. One group would design with the pre-conception, and another would design without it. The performance of the two groups would then be compared. The advantages of author practice stem from the use and acquisition of tacit knowledge about the pre-conception, and the difficulty of a zero-->build strategy. Tacit knowledge is important in two ways. First, tacit knowledge of pre-conception use may have been acquired during development of the pre-conception. Since designers from outside the research programme are without such knowledge, experimental subjects may find the pre-conception unusable. Without such tacit knowledge, given a basic representation, designers may be unable to derive, or make explicit, a suitable application representation and the methodological knowledge for using it. Designers without the benefit of an insider's knowledge of the pre-conception simply may not comprehend the pre-conception sufficiently well. For example, without implicit knowledge about the intended meaning of a poorly elucidated concept, a designer may find the pre-conception unintelligible and impossible to apply successfully. Given the need to develop and apply the pre-conception rapidly, it is reasonable to expect much of the pre-conception to be poorly formulated and somewhat unstable, so access to tacit knowledge may be particularly important. Second, tacit knowledge of the application representation that it is appropriate to develop, and how to use this representation may be acquired during the 'Conduct and Report Carry Forward Practice' workstream, and when the author of the pre-conception participates in such trials, such tacit knowledge may be retained within the research programme. For example, suggesting more correct and precise formulations of concepts, may speed the development of the pre-conception. The act of designing with the pre-conception may also reveal unexpected ways of reasoning with domain knowledge, and thus assist the explication of the relations between basic, application and design representations. Even if it were possible to train designers from outside the research programme to use the pre-conception, and devise an appropriate design problem for them to address, it is unlikely that any amount of protocol analysis and debriefing would elicit all their tacit knowledge of pre-conception use. All types of knowledge, however pure, consist, in part, of tacit rules which may be impossible to formulate in principle (Collins, 1982). Some aspects of knowledge constitute 'an ability to do something' and the rules that govern 'what may be done' with knowledge are often unstated (Winch, 1958). So large may be the proportion of tacit knowledge that, in some cases, such as technology transfer, knowledge has been regarded as a property of individuals rather than documents (Burns, 1969). At this stage in the research, tacit knowledge is too valuable to lose.

Another advantage of the author trialling the pre-conception is the difficulty of realising carry forward in informal Engineering by zero-->build (the task that this thesis has set itself). For example, any designer subjects would presumably be familiar with carry forward as conventionally practised, either as advice giving, or reasoning with heuristics. It is unclear, however, that randomly selected designers could learn to design in a radically different way, specifically, to abstract and instantiate general (class) requirements and specifications during the time allotted to an experiment. Indeed, to the extent that such knowledge representations and practices are novel, such experiments simply pass onto someone else the researcher's task of acquiring knowledge. In addition, given the lack of knowledge about effective combinations of basic, application and design representations and how to develop, use and increment them, it is unclear how such experiments could be designed. Specifically, how could problems for designers to address be selected if the appropriate type of problem was not known in advance? If a novel type of model is developed, there is no experience of using such models in support of design, and so little or no grounds for asserting *a priori* the application representations to develop or the aspect of general specification to support. It seems to be the responsibility of the author to find a purpose for any basic representation acquired. It is difficult to know, *a priori*, which characteristics of a pre-conception are, in principle, desirable or which carry forward practices, or innovations such characteristics enable. The need actively to seek out appropriate purposes given the knowledge a researcher possesses should not be under-estimated.

It may be argued that trialling the pre-conception in this manner is flawed in that any outcomes may be attributable to the individual involved, rather than the pre-conception, and that, interested parties, such as the author, may bias the outcomes obtained. Such arguments require the assessments of the case histories to be qualified and presented as weakly suggestive judgements, rather than 'conclusions', and to encourage the reader to form their own opinion about the case reported. Given the magnitude of the objective, however, even the opportunity to form an opinion now is of value, and a necessary preliminary to experimental studies.

1.2.4. Assessing the Basic Representation and the Reports of Carry Forward Practice: Comparative Review

In this thesis, assessment of the pre-conception and the case studies of carry forward practice are evaluated using the analytic technique of comparative review. That is, reflection upon the pre-conception and the case studies enables judgements to be made about the quality of the pre-conception, the effectiveness of the carry forward practice it supports. An analytic technique is required given the limited resources of a PhD and the requirements of the Zero-->build strategy. With analytic techniques, evaluation is possible, but its quality may be low, particularly if the 'expertise' supporting the evaluation is limited (Howard & Murray, 1987), and expertise in immature HCI Engineering is assumed to be low. Thus, the outcomes of such assessments are preliminary and suggestive only. In particular, analytic techniques may be subject to the bias inherent in human judgement and such basis is not always stated. In an attempt to make good some of the lack of expertise and potential biases, the reviews conducted are comparative. That is, both the pre-conception and the case studies are reviewed in the context of a characterisation of a the conventional, explicit Craft alternative. Conventional craft, here, is a distillation of the fundamental characteristics of a number of variations of current practices. Thus, the pre-conception is compared to end-users' domain knowledge (users are the conventional source of domain knowledge) and attempts to carry forward by instantiating classes are compared with carry forward by re-interpreting previous instances.

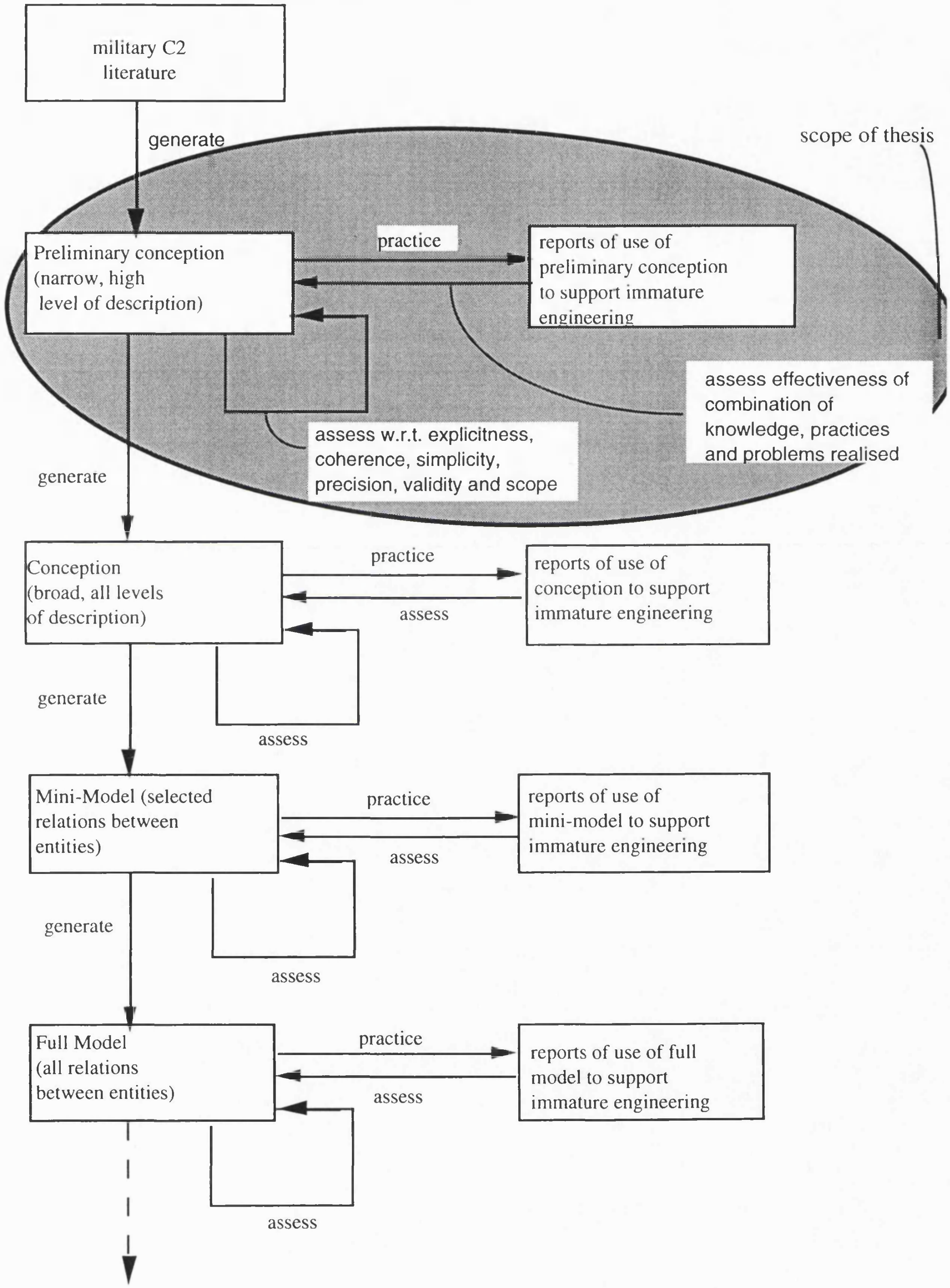
Assessments of the pre-conception by such 'quick and dirty' evaluation techniques does not suggest that the form and content of the pre-conception is unimportant. Indeed, sufficient explicitness, coherence, simplicity, accuracy and scope are necessary to enable the carry forward practices desired, although it may be difficult to state a priori how much is enough. For example, unclear or contentious content may be difficult or impossible to apply, or suggest ineffective designs, but, given little knowledge of the relations between domain knowledge of different types, and practices and problems, it is difficult to say in advance of designing whether or not a given lack of clarity is acceptable. However, it is necessary to guard against the intellectual reflex to perfect a model of C^2 before applying it, since such a reflex is incompatible with Zero-->build. Such a reflex would simply delay the realisation of informal HCI Engineering and evade the problematic issue of the model's purpose and the effectiveness of the carry forward practices it supported. It is necessary to emphasise the fact that well-formed models of perfect content are not required for all purposes. The pre-conception need be only coherent and correct enough for the purpose at hand. In the context of Zero-->building informal HCI Engineering, indiscriminate attempts to produce a well-formed model of C^2 with perfect contents may be wasteful and over-elaborate, since explicitness, accuracy etc. beyond the required level may not result in any additional benefit.

Assessments of carry forward practice may similarly be regarded as 'quick and dirty' since the same comparative review technique is utilised. However, constant use of such technique does not imply that, for this thesis, assessment is of little importance and conducted only a little. On the contrary, when regarded from a more conventional research perspective - that of generating basic representations alone, rather than a combination of basic, applications and design representations - this thesis appears to comprise more assessment that generation (see Figure 1.16). For example, the attempts to use the pre-conception to address design problems may be more conventionally construed as 'assessing the pre-conception's fitness-for-purpose' and thus the larger part of the thesis' resources has been devoted to evaluation. Indeed, the whole concern with knowledge-practice-problem relations reflects a desire to expose the pre-conception to assessment with respect to the important criteria - those which concern the relative effectiveness in practice of the way of conducting HCI that the pre-conception enables, and whether the pre-conception enables any progress at the discipline level and, if so, of what type.

1.3. Content of Thesis

What follows, then, is an attempt to realise, and then illustrate, the knowledge, practices and problems of informal HCI Engineering with particular respect to carry forward.

Figure 1.15: A 'Basic-Representation-Centric' View of Strategy for the Development of Human-Computer Interaction as Immature Engineering



The pre-conception is presented in Chapter 2. First, the domain of military *operations* is characterised and illustrated with reference to a particular military incident. Second, the domain of military *plans* is similarly characterised and illustrated. Finally, the pre-conception is assessed by comparing it to the conventional view of the domain - that acquired from users. With respect to military operations, the principal differences between conceptions are in terms of scope (the end-user's view tends to be wider) and level of description (the pre-conception is more abstract). With respect to military plans, the pre-conception appears to be more correct.

Chapter 3 reports the use of the pre-conception to support a late, summative evaluation of a reconstructed off-load planning system and compares the informal Engineering evaluation conducted with an explicit Craft equivalent. In the informal Engineering evaluation, carry forward involves the development, use and incrementation of, first, an expression of the planning system evaluation Problem, and second, a Problem hierarchy. In the explicit Craft evaluation, carry forward involves the selective re-use of views of the effectiveness of many other military planning systems, and the interpretation for the purposes of a view of effectiveness in general. Review of the case study suggests that, in this case, the informal Engineering evaluation addressed a range of HCI concerns and expressed effectiveness in a concise way. The informal Engineering evaluation was also different to the explicit Craft evaluation in terms of how the problem was conceived and how the completeness of the evaluation was reasoned about. Informal Engineering was partially realised in that an informal class of problem was formulated, and an instance of this class of problem was addressed. However, no class specification is apparent - just a highly selective instance specification. Carry forward was also partially realised in that a general view of planning effectiveness was derived from the pre-conception and used to develop a view of, first, an off-load planning system, and, second, a Surface-to-Surface Guided Weapons planning system. The Problem hierarchy was also incremented. However, re-use was not fully carried through. Overall, the informal Engineering evaluation was not judged by the author to be more effective in practice than the explicit Craft alternative.

Chapter 4 reports the use of the pre-conception to support the abstraction, specification, and specialisation of a generic menu structure for military planning systems, and compares the instantiation of this 'generic graphical user interface object specification with conventional, bespoke design. In the informal Engineering specification, the application representations involved are a Problem hierarchy, a general problem element, which concerns initiating planning interactions, and a domain-sensitive, generic graphical user interface object specification, that specifies a menu structure for military planning systems. First, a generic menu structure and its associated generic Problem element are developed, with reference to Graphical User Interfaces (GUIs) (the artefacts), analyses of GUIs, and an expression of the General Design Problem. Second, the generic menu structure and generic Problem element are specialised for military planning, with the support of the pre-conception of C^2. Guidelines for the design of menu structures are applied throughout. Comparison between the products of informal Engineering and the explicit Craft specification suggests that, in this case, the former is likely to result in interactions which are more consistent, which may benefit users who wish to interact with a number of different planning systems. Further assessment of the specifications were inconclusive (apart from highlighting the limitations of selected method), but failed to establish that the instantiations of the generic menu structure were *ineffective* relative to the bespoke menu structures of equivalent demonstrator systems. The informal Engineering specification was different to the mature Craft specification in terms of how the specification was conceived and how the completeness of the specification was reasoned about. Immature engineering was more fully realised in that a general specification was developed, and instantiated, albeit in a preliminary way. Carry forward was also more fully carried through, design work involved the development and instantiation of a class specification, and additional opportunities for carry forward were, in this case, created and exploited.

The challenge for future years is to devise and realise a manner of carry forward that is not only characteristic of immature HCI Engineering, but also more effective than the manner illustrated here, or achieved at full-scale and in the context of actual problems.

Chapter 2.
A Preliminary Conception of the Domain of Military Command and Control

Summary

This chapter presents a preliminary conception (pre-conception) of the domain of C^2. In Chapter 1, the pre-conception was characterised as a set of concepts for formulating the domain aspects of an informal, and general (class level) design Problem. This Problem involves military commanders and their staff interacting with C^2 equipment to plan and control military operations effectively. The pre-conception focuses on the abstract characteristics of tasks that are intuitively central to the notion of C^2 and takes the form of a component (part-of) hierarchy of domain objects, with partial attribute lists for each object.

The domain of C^2 is conceived as focussing upon two distinct, but mutually supportive kinds of work - military operations (armed-conflict) and plans for armed-conflict. Hence, two domains - armed-conflict and plans for armed-conflict - are analysed. The pre-conception is summarised below (see Figure 2.1):

The Domain of Armed-Conflict

Objects

STATE INTERESTS[1]: a plan for the use of resources.

ARMED-CONFLICT: FRIENDS and ENEMIES seeking to *secure* STATE INTERESTS by the display or use of force.

FRIENDS: objects whose INTERESTS support those pursued by the planning worksystem[2].

ENEMIES: objects whose INTERESTS are incompatible with those pursued by the planning worksystem.

[1] Strictly speaking, INTERESTS and their *security* characterise possible goals of planning work, and so comprise part of an analysis of the domain of plans for armed-conflict. However, because of the inseparability of planning and control (see later), these concepts are best introduced here.

[2] The planning worksystem, here, is that from which the conflict is conceptually inseparable.

NEUTRALS: objects whose INTERESTS are compatible with those pursued by FRIENDS and HOSTILES.
MILITARY SYSTEMS: a network of interacting human and technological elements which seek to *secure* INTERESTS by the display or use of force.
SYSTEMS: a network of interacting human and technological elements, which do not seek to *secure* INTERESTS by the display or use of force.
MEN: human beings.
EQUIPMENT: technological artefacts.

Attributes

Security: the potential of STATE INTERESTS to be realised.
Advantage: the potential a FRIEND or ENEMY to prevail in a CONFLICT.
Power: the potential of a FRIEND to gain *advantage* by the display or use of force.
Threat: the potential of an ENEMY to gain *advantage* by the display or use of force.
Vulnerability: the potential of a FRIEND, ENEMY or NEUTRAL to have harm done unto it by some means.
Involvement: the potential of a FRIEND, ENEMY or NEUTRAL to influence, or be influenced by, another FRIEND, ENEMY or NEUTRAL.
Fire-Power: the potential of a SYSTEM to deliver destructive force.
Safety: the potential of a SYSTEM to have its life endangered/be *damaged*.
Movement: the potential of a SYSTEM to change location.
Lift: the potential of a SYSTEM to transport MEN and EQUIPMENT.
Disruption: the potential of a SYSTEM to act as intended or designed.
Cohesion: the potential of a SYSTEM to act as a single, coherent set of elements.
Mortality: the potential of MEN to live.
Morale: the potential of MEN to act given the limitations of human will-power.
Fatigue: the potential of MEN to act given the limitations of human endurance.
Sustenance: the potential of MEN to meet physiological needs.
Surprise: the potential of MEN to have inaccurate expectations.
Damage: the potential of EQUIPMENT to satisfy the functional needs of other system elements.
Supply: the potential of EQUIPMENT to meet its own functional needs.
Acquisition: the potential of EQUIPMENT to have accurate information.

The Domain of Plans for Armed-Conflict

Objects

PLAN: a representation of goal states of domain objects and/or desired future behaviours of a control worksystem.

SUB-PLAN: a specification of lower level goal states of domain objects and/or desired future behaviours of a control worksystem.

Attributes

scope: the delimitation of *content.*

time_Scope: the period of time or 'time window' to which *content* applies.

object_Scope: the domain objects to which *content* applies.

behaviour_Scope: the conflict control worksystem behaviours to which *content* applies.

view: the expression and representation of *content* through use of symbols.

view_Type: the type of representation.

view_Content_Options: selections of *content* to be expressed in a representation.

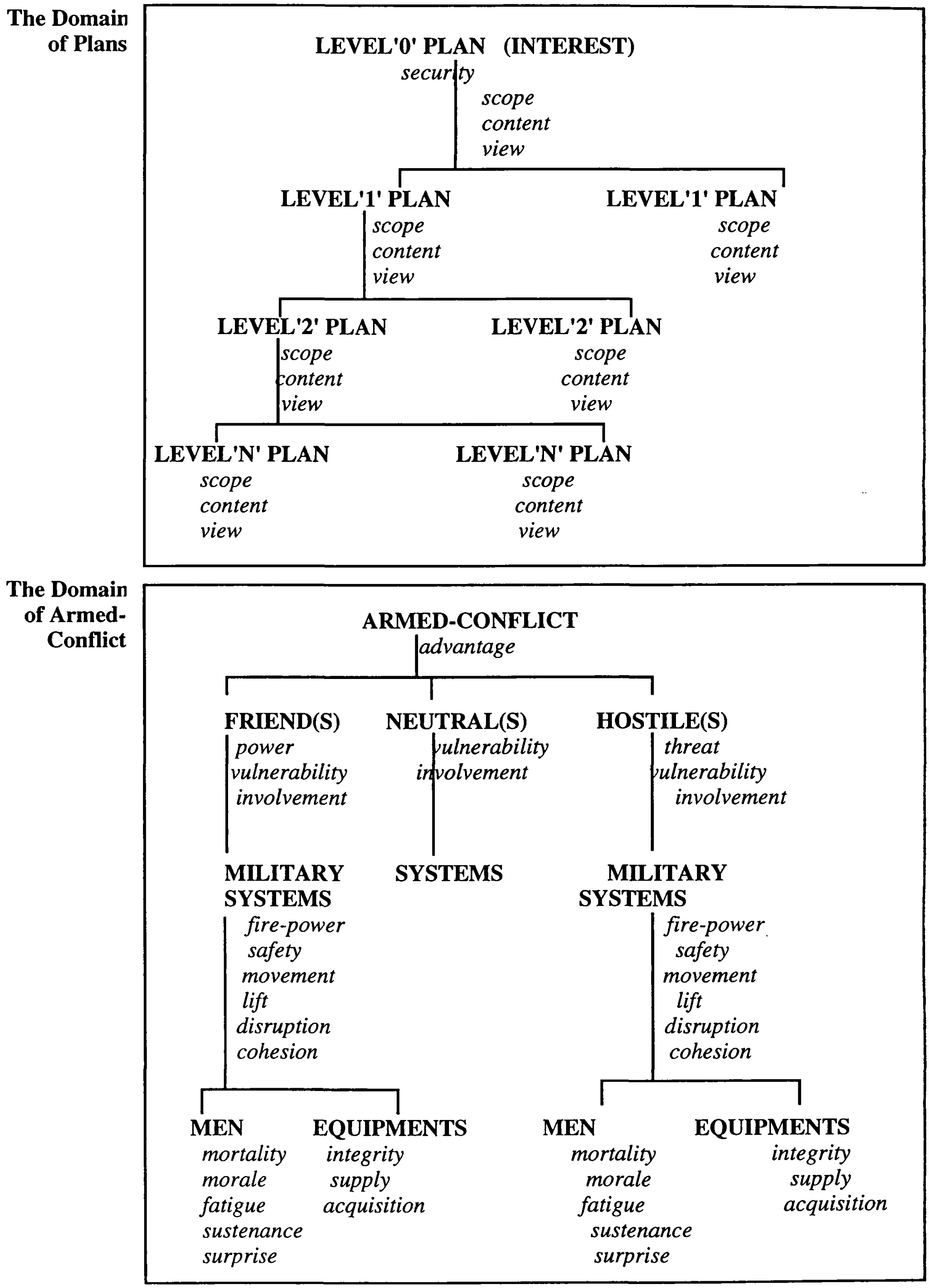
view_Format_Options: variations in the physical expression of *content.*

content: the specification of goal states of CONFLICT objects, and/or the behaviour of conflict control worksystems.

Some face validity for the pre-conception is claimed since, on initial inspection, at least, the pre-conception appears to be compatible with the consensus of domain experts (to the extent that such consensus exists). The pre-conception also appears to reference the same artefacts and events as end-users' knowledge of C^2. (End-user knowledge is presumed to be consensual, validated by successful performance of their tasks, and able to be made explicit without misrepresentation).

There are two sections in this chapter. The first section presents the concepts that comprise the pre-conception, illustrates them by modelling particular military incidents and relates the concepts to those of domain experts. The second Section compares the pre-conception with the end-user view of C^2.

Figure 2.1. The Planning and Control Domains of C2: Securing Interests Through Advantage in Armed-Conflict, and Specifying Goal States for Conflict Objects and Behaviours for Control Worksystems



2.1. Military C^2 as the Planning and Control of Armed Conflict

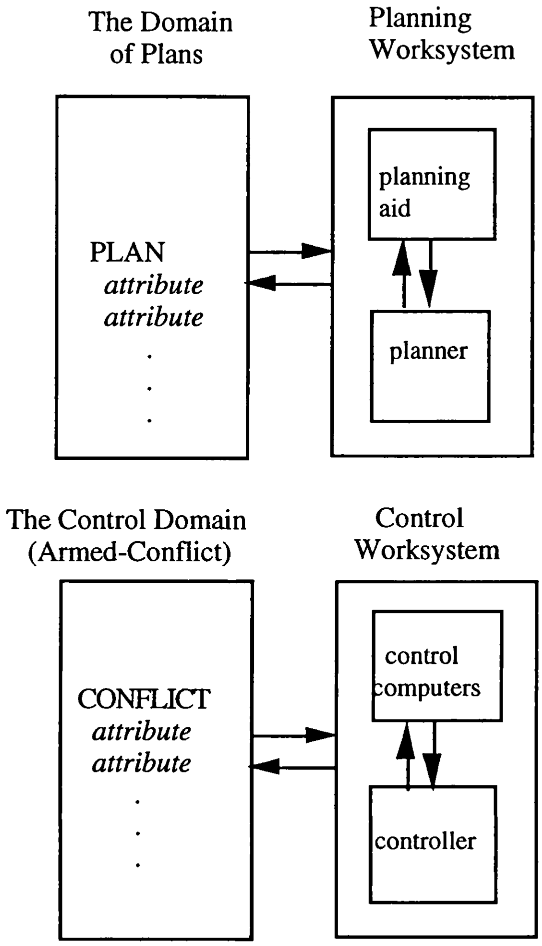
C^2 is conceived as a number of different, but complementary kinds of work (see Figure 2.2.). These kinds of work are mutually exclusive and supportive. Two domains within C^2 are distinguished - the domain of armed-conflict and the domain of plans for armed-conflict. Together, these domains constitute the management of armed-conflict. The domain of plans, here, does not refer to plans that may be better conceived as worksystem behaviours, for example, the conscious, or unconscious, thoughts of military personnel that might flash through their minds a few moments before they execute the control actions that the 'plans' guide. The domain of plans refers to what may be called 'preparedness plans' or 'operational plans', that is, possibly large, detailed and weighty documents, produced well in advance of their use, and whose production is required and monitored by some military organisation, for example, von Schlieffen's plan for the invasion of France and Belgium at the outbreak of World War I (Reason, 1991).

The view that C^2 comprises the control of military operations and the planning for them is not new:

"[C^2 is] 'the process of directing and coordinating military forces in the execution of the commander's will, and the exercising of his authority over all or part of the activities of sub-ordinate organisations"
Rice & Sammes 1989, p. 3

"Command, control, communication and intelligence is the management infrastructure for defence and war.... It is intrinsically a diverse range of activities."
Harris & White 1987, p. xi

Figure 2.2. A Conception of The Management of Armed-Conflict



Both military and cognitive analysts have suggested that a conception of planning is required for a conception of control, and vice versa.

"Plans are resources for situated action. ... Their purpose is to orient one in such a way that you can obtain the best possible position from which to use embodied skills."
Suchman 1987, p. 52

"Planning mechanisms incorporate two complementary functions that must be studied in conjunction. The first is plan elaboration ... The second is the use of plans. It is only by coordinating these two functions that an activity can acquire unity."
Hoc 1988, p. 7

"War is a mere continuation of policy"
von Clausewitz, 1832

"[War is] a form of political intercourse in which we fight battles instead of writing notes .... Everything that is strictly peculiar to military and naval operations relates merely to the means we use to achieve our policy"
Corbett 1911, p. 16

Planning and control, then, are inseparable. A plan is a means of making other individuals more able to control a military operation. A military operation is a means to an end, rather than an end in itself, and these ends are expressed through plans.

A conception of C^2 as the management of armed-conflict (planning and control) is the narrowest conception possible. A broader conception may conceive the domain of C^2 as, in addition, 'support for the management of armed-conflict' and so including, for example, a domain of 'pictures of armed-conflict' and the domain of 'communications about armed-conflict'.

The pre-conception of the domain of C^2 is presented in two sections. The first section conceives the domain of armed-conflict. The second section conceives the domain of plans for armed-conflict.

2.1.1. The Domain of Armed-Conflict

Concepts for characterising the domain of armed-conflict are first stated, then illustrated and, finally, reviewed.

2.1.1.1 Concepts

To clarify the conception, concepts proposed here are related to comparable military, and intuitive (lay) concepts whenever possible. Respects in which the pre-conception is narrow are also indicated. This indication may assist subsequent attempts to develop the pre-conception further.

The Security of State Interests

Security, in this conception, is the potential to realise State interests (see Figure 2.1). It is akin to intuitive notions of being 'firmly established', 'made fast' or 'guaranteed'. An interest is taken to be a plan for the use of resources. Such uses may be political, military, constabulary, or economic (Groves 1990; Booth 1977) and resources include the land, sea, air, space, and man-made installations or artefacts. Interests may also change in response to experience and may be more or less rational, precise, unified and stable (Weigly 1988). Realising an interest means bringing about the use of resources that was planned. For example, an interest may specify the transportation (a use) of oil by vessels of all nations through the Arabian Gulf (resources). Such an interest was expressed by the U.S. during the Iran-Iraq war of the late 1980s. Due to diplomatic and military efforts by the U.S. and other countries, this interest was, indeed, realised.

Security is conceived as pertaining to State interests, rather than personal interests. This distinction distinguishes armed-conflict from the criminal acts of individuals. A State, here, is an organisation for the pursuit of interests and comprises many sub- or interest groups (Claude 1988; Burton 1968). In the case of C^2, such interest groups include the Government, the military establishment, suppliers of military equipment and the general public. Further, security pertains to the interest that is the source, and objective of the conflict, rather than to any other interest. That is, security pertains to the plan that one side in the conflict asserts should be realised, and that the other side asserts, perhaps, implicitly, should not be realised. For example, in the case of the Iran-Iraq war, the U.S. wished the transportation of oil to be free from outside interference, whereas Iran wished the transportation of oil to be disrupted.

Security may take such values as 'secure', 'uncertain' and 'at risk'. Security may be indicated by people's expectations about the future, or the plans that are generally accepted to be realistic. For example, the price of a barrel of Brent crude oil (which is related to expectations about the balance between its supply and demand) may indicate the security of U.S. oil interests.

The concepts of interests and security make the inseparability of planning and control immediately apparent. To conceive the control domain of C^2 (the conflict) it is necessary to conceive, and reference, a domain of plans (the interests whose security conflict it is intended to achieve).

(Military)Power, Vulnerability and Involvement:
Having considered the 'ends' of a conflict (State interests), let us now consider the means employed to achieve them.

Military power is conceived, here, as the potential to secure interests by the display or use of force. It is the potential to damage or do physical harm to somebody or something with the sanction of the State, and is akin to the intuitive notions of 'potency' and of 'being empowered'. When referring to the power of an enemy, it may be more intuitive to refer to 'threat', rather than power, because of the tendency to view a conflict from the perspective of one participant or another - 'our side'. Power and threat, however, are essentially synonymous. Power is qualified by the term 'military', in order to distinguish C^2 from other kinds of work, such work as diplomacy, or trading, which may also pursue interests through power, but which involve power of different sorts, for example, political or economic. As an example of military power, a U.S. Marine Expeditionary Brigade notionally comprises around 26 ships, 17,000 men, 100 fixed wing aircraft, 100 helicopters, 90 vehicles, numerous anti-tank weapons, mortars and howitzers, and 30 days worth of combat supplies. By any standards, such a Brigade possesses considerable military power. As suggested previously, power may be exhibited by both friends and hostiles - those doing the fighting must be associated with some means of harming their opposition. Neutrals, in contrast, and by definition, may not necessarily be associated with any means of doing harm, and would not pursue their interests by exercising military power even if they possessed it.

Power may take such values as 'awesome', 'medium' and 'insignificant'. It may be indicated by the number, size and composition of units, their training, and the amount and type of equipment at their disposal. Power relates to the military concept of 'military capability', which is defined as 'the ability to achieve a specified wartime objective (win a battle, destroy a target set)'.

Power also relates to the military concept of 'power projection', particularly associated with operations outside a force's 'normal' area, for example, a force based in Western Europe evacuating nationals from an embassy in a remote part of the globe.

Military power is exhibited by one domain object (the 'seat' of power) towards another object - the one against which the power is directed (the subject of power, victim or target). If one object has the potential to damage another, then there must be some weakness on the part of the target that provides the opportunity for damage to be effected. The power of one object, then, implies the vulnerability of another. In this sense, vulnerability is the reciprocal of power. Vulnerability is conceived, here, as the potential to be subjected to force, or to be damaged, and is akin to intuitive notions of 'weakness' and 'Achilles heel'. For example, during an amphibious operation, a landing force is particularly vulnerable as it moves ashore. Its various elements are close together (rather than dispersed), at sea (where troops may drown) and in transit (the process of moving may limit a force's ability to defend itself). Vulnerability may be exhibited by friends, hostiles and neutrals - almost every participant in a conflict has the potential to be harmed in some way.

Vulnerability may take such values as 'impregnable', or 'highly vulnerable' and may be indicated by the number of missiles that must hit a ship before it would be 'dead in the water', or the number of weak-spots in a protective shield.

The power of one object, and the reciprocal vulnerability of another, implies the involvement of these objects. Involvement is conceived as the potential to unwillingly influence, or be influenced by, the conflict. It relates to intuitive notions, often applied to civilians, of 'being in the wrong place at the wrong time' and 'getting caught up in a conflict that does not concern someone'. Conflict is rarely isolated from the rest of human life. Consequently, there is often the possibility that an impartial, third party may unwittingly, or wittingly, become involved. Involvement may not only apply to unfortunate neutrals. It may also apply to circumspect friends and hostiles. All military units do not wish to get involved in all conflicts all the time. Decision concerning whom shall fight whom, and when and where, are crucial decisions and not all opportunities to realise power are necessarily taken. For example, consider a fighting force that is divided into two elements. One element may be ordered to engage enemy resistance, while the other may be ordered to avoid enemy contact and proceed to the objective. It is through the concept of involvement that, in the pre-conception, such events would be expressed.

Involvement may take the values 'intensely involved', 'involved' or 'uninvolved' and may be indicated by the number of enemy contacts reported per day, or the amount of ammunition consumed.

Friends, Hostiles and Neutrals

As suggested by the characterisation of power, vulnerability and involvement, the military means for securing interests concern three types of object - friends, hostiles and neutrals. These objects are active, intelligent and goal seeking. They are to some extent self-determining, that is, their state is not simply dependent upon interventions from a C^2 system or other domain objects. They have some freedom of action, and may adopt the state which offers the best opportunity to secure their intentions. Friends, hostiles and neutrals may be distinguished according to the objective that they are pursuing, relative to the interests pursued by the C^2 system whose domain they comprise. Friends pursue interests that are identical to, or intentionally supportive of, those pursued by the C^2 system. Hostiles intentionally pursue interests that are incompatible with those pursued by the C^2 system. That is, they may seek uses of resources that are incompatible with those sought by the friend, or which obstruct progress towards the use desired by the friend. Neutrals, in contrast, pursue interests that are compatible with those pursued by both friend and hostiles. Consequently, they have an attitude of impartiality towards the conflict. For example, a C^2 system may seek to secure the interest of sovereignty over a particular territory. Friends, the force that is under the command of the C^2 system, may seek to clear other nations' forces from this area. Hostiles, the forces of other nations, may seek to stand their ground. A neutral may wish to fly over the territory whose sovereignty is disputed, in order to reach its destination, and not in order to influence which of the adversaries succeeds in the land war.

Thus, this conception of the domain of C^2 distinguishes friends, hostiles and neutrals relatively clearly. It may be objected that during a conflict, and from the point of view of military commanders, there may be considerable ambiguity about the identity and intentions of relevant objects. Difficulty is classifying an object as a friend, hostile or neutral in a particular instance frequently leads to problems and mistakes, such as attacks by one unit against other on its own side, and are part of what has been called 'the fog of war'. But ambiguity about the status of a conflict object is in the worksystem, not in the domain. Just because the knowledge that supports certain C^2 system behaviours may be limited or inaccurate does not necessarily prohibit the unambiguous definition of the classes of object that comprise conflicts.

Advantage in Armed-Conflict

It has been proposed that the military means of securing interests comprises friends, hostiles and neutrals, and their power, vulnerability and involvement. Taken together, these objects and attributes comprise an armed-conflict. This conflict may be on any size or scale, from actions and engagements, to battles, campaigns and wars. Previous references to State interests may have suggested conflict on a large scale, and in the pursuit of strategic interests expressed by high-ranking, government representatives. However, smaller, specific interests may also be expressed on behalf of the State, perhaps, by military commanders, and smaller, more limited conflicts may be conducted in pursuit of these interests. For example, a Brigade commander may specify the use of a hill (a resource) for observing enemy activity (a military use) and may seek to secure this interest by means of an action involving one or two Platoons.

In a conflict, friends and hostiles compete for military advantage, that is, a favourable, relative potential to influence security by military means. Advantage, then, expresses the matrix of power and vulnerability relations that exists between the participants. It expresses the discrepancy between the two sides' total faculties for prevailing upon each other in that situation. Advantage relates to intuitive notions of 'competitive advantage', 'having the edge', 'initiative' or 'upper-hand'. It is loosely comparable to what, in International Relations, is referred to as 'the balance of power'. Whereas the concept of 'balance of power' emphasises mutual security through maintenance of approximately equal capabilities, 'advantage' emphasises the achievement of whatever level of security is desired.

Military advantage may take such values as 'in one side's favour', or 'evenly balanced'. A related military concept is 'strategic advantage', which is defined as 'the overall relative power relationships of opponents that enables one nation or group of nations effectively to control the course of a military/political situation'. When a particularly high level of advantage is obtained, one side may effectively determine what occurs within a certain geographical area of the conflict. Under such circumstances, military terms such as 'having area control' or 'having air superiority' may be appropriate. Advantage may be indicated by the number of military objectives achieved by each side, or which side is judged to have the initiative.

The domain of C^2, then, may be characterised as the securing of State interests through advantage in armed-conflict.

<u>Military Systems, and Their Attributes</u>

Having conceived the domain of C^2 at a high level, let us now consider it in more detail. Friends and hostiles are conceived as comprising one or more military systems. A system, here, is taken to be a socio-technical system comprising a number of interacting elements. Military systems may be distinguished from other systems in terms of the attributes that they exhibit and the fact that they seek to increase power. Military systems are taken to exhibit fire-power, safety, movement, lift and disruption. Fire-power, here, is the potential to deliver destructive force. Such force may be physical, chemical, biological or nuclear. Fire-power may be indicated by the number of weapons that a friend or hostile possesses, the technical sophistication of these weapons, their effective range, or rate of fire. Fire-power is also a military concept, and is defined as 'the amount of fire which may be delivered by a position, unit or weapon system'. Safety, in this conception, is used in its intuitive sense - the potential to have one's life endangered. Movement is conceived as the potential for changes of position. Position, here, implies not just physical location, but also orientation and formation. Reference to movement rather than expedition implies that, during military operations, financial considerations are subordinated to tactical objectives and requirements. (Expedition is taken to imply change in location for some financial cost, and may be more applicable during exercises or administrative operations). Movement may be indicated by top-speed, rate of turning, engine capacity or thrust. A related military concept is 'mobility', which may be defined as 'a quality or capability of military forces which permits them to move from place to place while retaining the ability to fulfil their primary mission'. Lift is the potential to transport or carry other objects, that is, to load, move and unload. It relates to the intuitive notions of 'giving a ride' and 'carriage' and may be indicated by maximum recommended capacity. Finally, disruption refers to the potential to frustrate or make difficult the normal functioning of socio-technical systems without endangering life. It is particularly associated with non-destructive counter measures, such as decoys and jamming and may be indicated by the number of decoys in a field of view, or the number of jammed radio frequencies.

Men and Equipment, and Their Attributes

Generally speaking, military systems are socio-technical systems, comprising humans and technology working together. In the pre-conception, human beings, are referred to as men and the technology is referred to as equipment. Men may exhibit a variety of attributes, including mortality (the potential to lose one's life), morale (the potential to act willingly), fatigue (the potential to act without rest), sustenance (the potential to satisfy the needs of day-to-day human existence - eating, drinking and keeping warm), cohesion (the potential for individuals to act as a single, unified body) and surprise (the potential to be caught unawares). Equipment may also exhibit a variety of attributes, including damage (the potential to function without repair or maintenance), supply (the potential to consume the materials necessary to function, such as fuel and ammunition) and acquisition (the potential to acquire information). Acquisition, here, is comparable to surprise, but is applicable to equipment, such as missiles or sensors, rather than to men.

A broader conception may postulate additional concepts. For example, it may postulate an environment which contains the conflict, that is, the natural environment of earth, sea, air and space and its interfaces, such as the coast, sea-floor and sea-surface, and land-surface. It may also attempt to capture the full variety of military technology. For example, equipment that jams radio-frequencies may exhibit an 'interference' attribute. A broader conception may also postulate additional levels of abstraction. For example, it may be possible to distinguish different types of military system according to the functions that are performed. For example, combatant objects, which actually do the fighting, may be distinguished from logistics objects, which support the combatants.

2.1.1.2. Illustration: Accidental Downing of Iran Air Flight 655 on May 3rd, 1988 by U.S.S. Vincennes

In this section, the concepts outlined above are illustrated by modelling an actual conflict control task concerning anti-air warfare and sea control - the accidental downing of Flight 655 by the U.S.S. Vincennes in 1988. Because of the nature of the pre-conception, the model reflects a narrow view of the conflict and describes its abstract features. Further, because the pre-conception provides only domain knowledge, only tasks are modelled. There is no attempt to model the command and control behaviours that bring about these tasks.

The Iran and Iraq war of the early 1980s was initially limited to a land battle. Iraq wished to disrupt trade in oil between Iran and the outside world, and so bring further economic pressure to bear on an already financially weak Iran. In pursuit of this aim, Iraq launched numerous air attacks against Iranian oil installations. Iran's response was to disrupt the transportation of oil through the Persian Gulf. This disruption, it was hoped, would provoke Western diplomatic pressure on Iraq to suspend its own attacks on Iranian oil assets. To ensure the free transportation of oil, U.S. naval forces were sent to the Gulf. During the late 1980s, U.S. forces were involved in a number of incidents and actions. The more notable of these incidents include: (i) in May 1987, an accidental, Iraqui air attack on the U.S. frigate 'Stark' killed 37; (ii) also in May 1987, a U.S.-flagged Kuwati tanker Bridgeton hit a mine (the U.S. blamed Iran); (iii) in April 1988, an explosion holed the U.S. frigate Samuel B. Roberts (the U.S. blamed an Iranian mine); and (iv) the U.S. demolished two Iranian oil installations and six Iranian patrol boats. The Vincennes incident occurred on July 3rd, 1988. USS Vincennes, a U.S. cruiser, accidentally shot down a commercial aircraft en route to Dubai with the loss of 298 lives (US DoD 1988). The events of July 3rd, 1988 are described in Table 2.1. The left hand column describes events in end-user (military officer's) terms. The right column (the model) describes the task in terms of concepts from the pre-conception. The model focuses on the power, vulnerability and involvement of the friends, hostiles and neutrals concerned, namely three friends (Vincennes, Montgomery, and a Pakistani oil tanker), an enemy (a group of seven Iranian small boats) and a neutral (Iran Air Flight 655). The tanker is considered to be a friend, rather than a neutral, because its interest is not compatible with the Iranian interest, and is supportive of the Americans'. Note that all of the information contained in the end-user description is expressed technically. Also, the friend 'Vincennes' refers to all military systems that comprise this fighting unit, and so 'Vincennes' includes not only the ship itself, but also the ship's helicopter. Some aspects of the model are summarised in Figure 2.3. For clarity, domain objects are in capitals, domain attributes are in italic.

It is the morning of July 3rd, 1988. In the previous week, the Iraqui Air Force had successfully attacked Iranian forces near the north Persian Gulf. Iranian retaliation in the form of massed small boat attacks on commercial shipping was expected.

Table 2.1:A Conflict Control Task as Performed by USS Vincennes' C^2 System on July 3rd 1988

End-user Account	Model
0330 USS Montgomery reports that approximately seven Iranian small boats have approached and challenged a merchant ship (the tanker). Some explosions are heard.	The *threat* of the ENEMY (the small boats) with respect to the RESOURCE (tanker) increases. The ENEMY makes some attempts to realise its *power*. The *security* of the U.S. INTEREST is reduced.
0412 The Vincennes is ordered to the area to support USS Montgomery and investigate reports of small boats challenging a tanker.	The FRIEND's (Vincennes) *power* and *involvement* increase, and the *vulnerability* of the ENEMY increases accordingly.
0610 The Vincennes' helicopter is diverted from a routine patrol to reconnoitre the area of the small boats. By so doing, the helicopter and the small boats come within range of each other's weapons.	The *involvement*, *power* and *vulnerability* of one of the FRIEND's MILITARY SYSTEMs (Vincennes' helicopter) and the ENEMY increase rapidly.
0615 The helicopter is fired upon by the small boats, but no damage is inflicted. Having established the small boats' intentions, the helicopter returns to the Vincennes.	An attempt to realise the *vulnerability* of one of the the FRIEND's MILITARY SYSTEM fails. Levels of *power*, *vulnerability* and *involvement* reduce.
0620 The small boats and the Vincennes continue to close.	The *involvement*, *power* and *vulnerability* of the FRIEND (Vincennes) and the ENEMY increase again.
0643 The small boats and the Vincennes continue to close. Two small boats turn towards the Vincennes, while the other small boats manoeuvre erratically. Note that the small boats have been drawn away from the tanker.	The *involvement*, *power* and *vulnerability* of the FRIEND (Vincennes) and the ENEMY continue to rise. (The FRIEND's *vulnerability*, however, is relatively low.) The *threat* of the ENEMY with respect to the FRIEND (tanker) is reduced. The U.S. INTEREST appears more *secure*.
0647 Flight 655 takes off from Bandar Abbas, an airfield near the North coast of the Gulf and used by both civilian and commercial aircraft. An air contact is detected by the Vincennes, and designated 'unknown, presumed hostile'.	The involvement of the 'neutral' (Flight 655) and the FRIEND (Vincennes) begins to increase.

<u>0649</u> Flight 655 adopts its routine flight path, which is, by chance, towards the Vincennes. The Vincennes challenges its air contact (actually Flight 655) but receives no reply. The period immediately after take-off is a busy time for flight crew, so they may not have been monitoring Air Distress frequencies. For a fleeting moment, Vincennes' air contact appears to electronically identify itself as a military aircraft. (This was probably due to the prevailing atmospheric conditions). Flight 655 is approaching the range of Vincennes' surface to air missiles.	The *involvement* and *vulnerability* of the NEUTRAL and the FRIEND's power with respect to it, increases rapidly.
<u>0651</u> One of the Vincennes' guns jams at a moment when one of the small boats is about to adopt a potentially dangerous position. A sharp change in course brings the remaining gun to bear on the small boat posing the greatest threat. Further challenges to the air contact receive no reply. Flight 655 is well within range of the Vincennes' surface to air missiles. Flight 655's altitude is mis-read. Flight 655 is perceived as diving towards the Vincennes, whereas it is, in fact, climbing away from it.	The *power* of the FRIEND with respect to the ENEMY temporarily falls sharply, but is soon restored. The *vulnerability* of the ENEMY fluctuates accordingly. The *power* of one of the ENEMY's MILITARY SYSTEMs (a small boat) also rises slightly before falling back. The FRIEND's *vulnerability* fluctuates accordingly. The *involvement* and *vulnerability* of the NEUTRAL and the FRIEND's *power* with respect to it continue to increase rapidly.
<u>0654</u> Two surface to air missiles are launched by the Vincennes towards its air contact, in the belief that the ship is under attack from an ENEMY fighter. In fact, the missiles hit Flight 655, which is destroyed.	The FRIEND's *power* with respect to the NEUTRAL is realised, with catastrophic results.
<u>0703</u> The small boats begin to leave the area. It is confirmed that one small boat has been destroyed and that the Vincennes has incurred superficial damage as a result of small arms fire or shrapnel. The Vincennes learns that is has mistakenly shot down a commercial aircraft.	The *power* of the ENEMY (with respect to the FRIEND) reduces, as does the *involvement* of all parties.

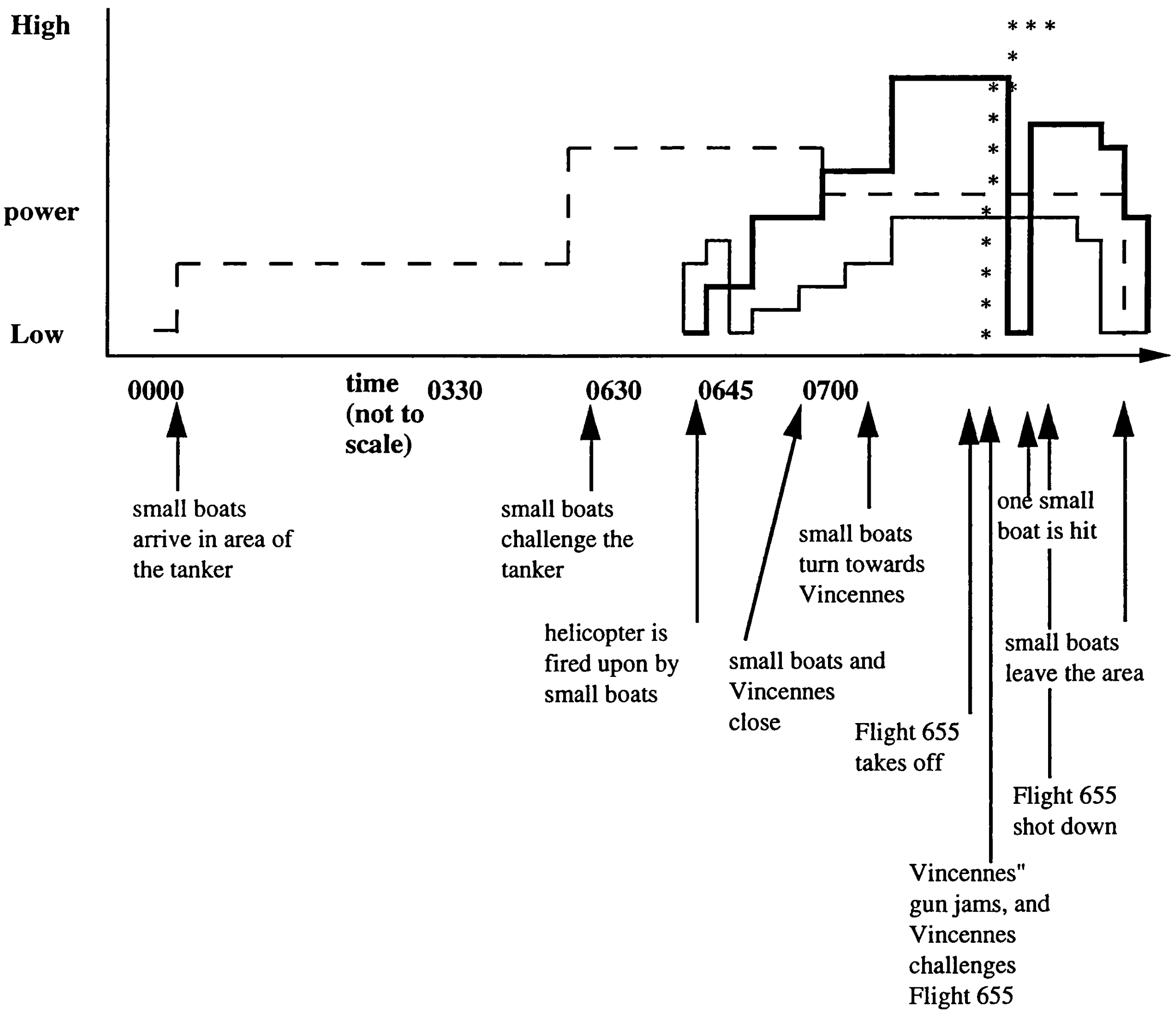
Figure 2.3: A Model of a Conflict Control Task: The Vincennes Incident

Power of the ENEMY (small boats) with respect to the FRIEND (Tanker) (— —)

Power of the FRIEND (Vincennes) with respect to the ENEMY (small boats) (—)

Power of the ENEMY (small boats) with respect to the FRIEND (Vincennes) (—)

Involvement and *Vulnerability* of NEUTRAL (flight 655) (***)



The Vincennes incident, as was said at the time, 'gives a stronger voice to the radical elements [in Tehran] who do not want a compromise with the West. It may also become a useful rallying point for a regime which has suffered a series of recent losses against Iraq' (London Times, July 4th, 1988, p. 6). Also, 'hopes that the tanker war was winding down have been dashed .... the situation is now more perilous than ever' (London Times, July 5th, 1988, p. 12). However, the oil price (an indicator of the security of the US interests) was apparently unaffected. Traders were more concerned with the possibility of continuing wrangles within OPEC and, as this possibility receded, the chances that major buyers of oil previously absent from the market would return.

This review concludes the pre-conception of the domain of armed-conflict. Let us now consider the domain of military plans.

2.1.2. The Domain of Plans for Armed Conflict

2.1.2.1. Concepts

The domain of military plans comprises a single type of object, a military plan (see Figure 2.1). A military plan, here, is conceived as a representation of desired future states of conflict objects (friends, enemies etc.), and/or behaviours of a system that controls military operations (a conflict control worksystem). At the highest level, there is a single plan, a Level '0' plan. In the fore-going analysis of armed-conflict, this plan was referred to as the interest. A level '0' plan comprises a number of constituent plan-parts - level '1' plans. Level '1' plans, in turn, comprise a number of level '2' plans, and so on, until the most elemental 'plan-part' is reached.

Plans have at least three types of attribute - scope attributes, content attributes and view attributes. Scope attributes delimit the range of possible plan content. Time_Scope delimits plan content with respect to time, that is, a period of time to which plan content applies. Object_Scope delimits plan content with respect to the domain objects for which plan content specifies goal states. Behaviour_Scope delimits plan content with respect to the behaviours of conflict control systems that plan content may specify. Scope attributes define what a plan may be about and implicitly distinguishes the type of plan to be produced from other types of plan. For example, a logistic plan may have a scope attribute value of 'the movement of all friendly transport vehicles within X Group between 1/4/89 and 8/4/89'. Content attributes specify particular values within the range delimited by plan scope and so specify goals and/or behaviours for control work. Content is the meaning that a plan attempts to communicate. Whereas scope attributes delimit the range of possible content, content attributes nominate actual values within this range. For example, a logistic plan may specify that the friendly transport vehicles should achieve a rate of lift of 500 tons/hr for the first day of the operation. View attributes define the class(es) of physical attribute exhibited by the plan, that is, its potential to express meaning by the use of symbols. View refers to the form of the plan. View_Type expresses the type of representation, such as 'graph', or 'text'. View_Content_Options expresses the selection of content that is to be expressed in a particular representation. View_Format_Options expresses the particular variation within a type of view of content. that is to be adopted in a particular instance. For example, view attributes express the intuitive notions that a plan may be represented in a 'tabular' or 'graphical' form, but without showing airborne forces on this occasion, and highlighting heavy weapons, such as tanks and artillery.

Some attribute values are unique. For example, the content attributes of a plan may specify either a lift of 500 tons/hr or 600 tons/hr but not both. Other attributes may exhibit multiple values. For example, the view attributes of a plan may indicate that off-load is represented as a table, a gantt chart and a map at the same time.

Concepts of hierarchical plan decomposition, and scope content and view, then, comprise this narrow pre-conception of the domain of military plans. A broader conception may postulate additional concepts. For example, effect attributes may be conceived as expressing the 'added value' of the plan relative to an alternative, that is, another plan or no plan at all, the difference between specified and alternative possible future realities. Added value, here, refers to the difference between the way the conflict unfolds with the plan, compared to the way the conflict could have unfolded. Whereas content expresses 'the plan on paper', effect attributes express a plan's impact on what actually happens, relative to what could have happened. For example, in the previous example, scheduling the movement of transport vehicles may ensure their most efficient use. Without such scheduling, a rate of lift of only 350 tons/hr may be achievable. With the plan, this figure may rise to 450 tons/hr. In addition, a plan may be conceived as exhibiting status, in order to reflect the authority that lies behind the plan, such as 'pending', or 'approved by the commanding officer'.

The inseparability of planning and control (at least at a high level of description) is also apparent in this analysis of the domain of plans. In order to conceive scope and content attributes for military plans, it is necessary to have previously conceived armed-conflict (the control domain). For example, the scope and content of the logistic plan was expressed in terms of friends (transport vehicles).

2.1.2.2. Illustration: Planning an Hypothetical Amphibious Landing

Off-load planning is planning for the unloading of men and equipment during an amphibious landing. An amphibious landing may be considered to be an attack against a hostile shore, launched from the sea and involving air, sea and land forces. It involves the movement ashore of a landing force, embarked on transport ships and naval vessels, by means of amphibious vehicles, landing craft and helicopters. The landing force arrives ready for combat ashore, and at beaches and landing zones, rather than ports and airfields (Evans 1990; see also Thompson 1986 for an account of landings in San Carlos Water, 1982). Such operations are meticulously planned. A landing force is particularly vulnerable as it moves ashore and such movement involves the coordination and synchronisation of large air, sea and land forces. Planning focuses on the development of off-load tables. These tables include the Load&Stow, a tactical plan and a logistic plan. The Load&Stow specifies where equipment is to be stowed on transport ships, and the order in which the ship is to be loaded. The tactical plan specifies who is to go ashore, where, and when. The logistic plan specifies who is to take the landing force ashore and how the landing force is to be grouped as it moves. One line in the logistic plan corresponds to one load, either for an assault craft or a helicopter.

The off-load tables support the control of the landing on the day. For example, a helicopter controller may monitor progress by crossing-off loads as they leave the flight deck and keep track of the situation by marking any improvisations or deviations from the plan that occur as the operation unfolds.

The off-load tables also support the planning of other aspects of the operation. For example, naval gunfire support must be timed to cease just before the first waves of the landing force arrives on the beach. The gunfire support team must be informed about the intended location of friendly forces.

The following section describes hypothetical events in end-user terms (after Gaye & McCubbin, 1991) and in terms of concepts from the pre-conception. The example involves one government statement (the interest), a strategic response (level '1' plan) and a number of sub-plans, which are iteratively developed and selected in response to new information and events.

Let us suppose that there has been a coup on a remote pacific island, and the new ruler has vowed to 'sweep into the sea' the representatives of the Western Governments that sustained the previous, oppressive regime.

The landing is conducted the following day.

This concludes the pre-conception of the domain of C2. The following section presents an initial assessment of its face validity.

Table 2.2. An Hypothetical Planning Task as Performed by a Possible, Future C^2 System

End-user Account	Model
H (the time of the coup) +1 day The British Government issues a statement indicating that it regards the safe evacuation of its diplomats and businessmen as essential. The coup is not expected to last long, and a working British presence must return to the island as soon as possible.	An INTEREST (LEVEL '0' PLAN) is declared.
H + 2 days The Chiefs of Staff configure an amphibious force from available military units and shipping taken up from trade and propose a Services Protected Evacuation for the island. They also outline a general strategy for the brief campaign. The proposal is accepted.	A LEVEL '1' PLAN is produced.
H + 4 days The Staff of the Amphibious Brigade charged with performing the Evacuation, generate some alternative tactical options for amphibious landings at different parts of the island. In the light of these options, a plan for loading and stowing men and equipment aboard the Force's transport ships is drafted. (Equipment likely to be unloaded first must be loaded so that it is easily disembarked, e.g. near the doors). The amphibious force embarks and steams toward the island.	A number of LEVEL '2' PLANS are produced, together with a LOAD&STOW (a type of LEVEL '3' PLAN).
D - 9 days. The amphibious force puts to sea. Whilst en route to the island, intelligence reports about the rebels' disposition are received and a preferred tactical plan is finalised.	The *content* of the preferred LEVEL '2' PLAN become 'complete'.

<u>D - 4 days 09:00 hrs.</u> Two alternative plans are generated for the logistic aspects of the landing. Each options comprises a surface assault schedule for off-load by surface means and a helicopter employment and assault landing table for off-load by air. In one option, the transport ships are positioned well out to sea. In the other, the ships are positioned closer to the shore. Thus, the options differ in terms of the routes to be plied by landing craft and helicopters. To save time, only the timed waves (the initial waves, whose precise time of arrival is important) of the plan are produced. All the plans take the form of landing tables	Two types of LEVEL '3' PLAN are generated. The *content* of one set of LEVEL '3' PLANS is 'long trips'. The *content* of the other set of LEVEL '3' PLANS is 'short trips'. The *scope* of both sets of LEVEL '3' PLANS takes the value 'timed waves'. Their view takes the value 'table'.
<u>D - 4 days 12:30 hrs.</u> To assist the selection of a preferred option, the Staff represents each logistic plan as a gantt chart and as an animated diagram (in which the landing is seen to unfold from the enemy's point of view). A decision is taken to pursue the 'near-in' option, and the table, gantt chart and animated diagram are completed for this option.	The LEVEL '3' PLAN 'S *view* takes the value 'table, gantt and animation'. The *content* of the preferred option becomes 'complete'.
<u>D - 4 days 16:00 hrs.</u> Additional logistic plans are developed. These plans include the cross-decking plan, which specifies the movement of men and equipment to the ship from which they are to disembark, if they are not aboard this ship already, and the flight plans that specify, for each assault craft and helicopter, their activities on the day of the landing.	Additional LEVEL'4' PLANS are produced.
<u>D - 1.</u> Two helicopters are damaged in a storm and the logistic plan is revised accordingly. Local intelligence suggests that the storm has also disrupted water and electricity supplies at the rebels' headquarters. The Commander of the Amphibious Force decides to press on with the landing. Other plans are not changed.	The *content* and *scope* of the favoured LEVEL '3' PLANS are revised. The LEVEL '2' PLAN and the LEVEL '4' PLANS are unchanged.

2.2 Assessment of Face Validity

2.2.1. Compatibility with Consensus of Domain Experts

This assessment claims at least some face validity on the basis the pre-conception reflects the consensus of domain experts.

2.2.1.1. Essential and Distinguishing Features of Armed-Conflict

In agreement with what may be taken to be the consensus of domain experts, the pre-conception of the domain of armed-conflict presented here implies that certain features of C^2 work are essential and distinguishing. First, the Vincennes' actions were a means to an end, rather than an end in themselves (Hoc, 1988; Harris & White, 1987; Suchman, 1987; Corbett, 1911, von Clausewitz, 1832). The pre-conception reflected this assertion by conceiving of planning and control of military operations as inseparable activities. For example, power, an attribute within the domain of armed-conflict, may only be understood with reference to security - an attribute in the domain of plans. Interests, an object in the domain of plans, may only be understood with reference to systems - objects in a domain of armed-conflict. Second, in military operations, ends are achieved by displaying and using force (Booth, 1977; Groves, 1990). This aspect distinguishes the Vincennes' actions from other activities, such as diplomacy or financial inducement. The pre-conception expressed the violence of the incident within the notions of power/threat and vulnerability. Third, military operations are not simply a 'clash of armies', in which each side hurls itself against its opponent (Corbett, 1911, p. 11). Rather, the decision of whom should fight whom, and when and where, were crucial decisions, and all opportunities to engage the enemy were not necessarily taken (witness the withdrawal of the Vincennes' helicopter). The pre-conception expressed this characteristic of the incident within the concept of involvement. Fourth, military operations are not always isolated from the rest of ordinary life and non-combatants may intentionally, or unintentionally, be drawn into conflicts. The pre-conception expressed this characteristic of the Vincennes incident by distinguishing neutrals from friends and enemies.

To the extent that the essential and distinguishing features of planning work implied by the pre-conception reflect the consensus among domain experts, that is, the concepts accord with basic, published texts in the area, the pre-conception may be said to have at least some face validity.

2.1.1.2. Essential and Distinguishing Features of Planning Armed Conflict

The pre-conception of the domain of plans presented here implies that certain features of C^2 work are essential and distinguishing. First, the plans produced by the Brigade Staff are resources for control work rather than specifications of what to do next (Suchman, 1987). There is little expectation that military plans will be followed to the letter.

"Gentlemen, do not be dismayed if chaos reigns and these plans go awry, because chaos undoubtedly will reign"
Brigadier James Hill, on the eve of the Normandy landings, 1944
quoted in Thompson, 1986

This view of plans as resources for control also distinguishes planning from design work, since the latter is taken to concern resources for implementation (manufacturing, building etc.). Second, planning for the Evacuation followed the declaration of an interest, and involved the production of strategic, tactical, and logistic plans. These plans are hierarchically organised. The logistic plans form part of the tactical plans. The tactical plans in turn form part of the strategic plans. Each plan produced may be regarded as a 'partial' plan, in the sense that a plan incompletely specifies the subsequent operation and so calls upon plans lower in the hierarchy to make the additional commitments required for completeness (Currie & Tate, 1991; Chandresekaran et al., 1989). Further, a particular planning task may not necessarily require the parallel development of all levels of plan. Particularly in military planning, the development of lower level plans may be delayed until the best possible information is available (Applegate et al., 1990; Loberg et al., 1986). Third, the Chiefs of Staff and Brigade Staff preparing for the Evacuation were conceived as delimiting, representing and generating content about desired, future state(s) of some aspect of the military operation. This view of planning distinguishes it from related military tasks, such as intelligence gathering and picture compilation, which specify previous, or actual states of a conflict, rather than future states. It also distinguishes planning from the related work of personal information management, which is concerned with the availability of the content generated, rather than the content itself. That is, planning is distinct from librarianship and security activities that deal with the duplication, storage, retrieval and distribution of the plans produced. Further, planning as conceived here accommodates tasks in which a single future state is specified for a certain point in time (a plan as time slice), tasks in which a future state is specified for a period of time (a plan as a schedule) and tasks in which multiple, alternative future states are specified either for a point in time, or over a period of time (a plan as contingency plan or partial plan).

To the extent that the essential and distinguishing features of planning work implied by the pre-conception reflect the consensus among domain experts, , that is, the concepts accord with basic, published texts in the area, the pre-conception may be said to have at least some face validity.

2.2.2. Identity of Points of Reference

In this section, additional attempts are made to assess the face validity of the pre-conception. This assessment claims at least some face validity on the basis of identical points of reference with an end-user's conception of the domain. End-user knowledge is presumably validated by successful task performance so, by extension, apparently similar knowledge would be presumably valid also. The first assessment compares and contrasts alternative accounts of military incidents. The second assessment compares and contrasts alternative views of military plans.

2.2.1. Alternative Accounts of Military Incidents

A comparison of the left and right hand columns of Tables 2.1. and 2.2., suggests that there are differences between the end-user and pre-conception supported account of C^2 work in that the end-user account reflects an end-user perspective, that is, an individual 'doing the job', the human element of a C^2 worksystem, and is expressed in relatively concrete, operation-specific terms. The pre-conception account, in contrast, maintains distinctions asserted in Dowell & Long (1989), for example, between domain and worksystem, and addresses only the domain, and is expressed in more abstract, general terms. For example, with respect to the Vincennes incident, in the end-user account, the fire directed towards the helicopter, the manoeuvring of the small boats, and the approach of Flight 655 (domain issues) are related alongside the reception of freak IFF signals and the mis-reading of altitude information (worksystem issues), and expressed in the specific terms of anti-air warfare and commercial air and sea transport. The end-user account reflects how an individual, the human element of a C^2 worksystem, may experience military work. The pre-conception supported account, in contrast, expresses only the former (fire, manoeuvring and approach) and as fluctuations in power/threat, vulnerability and involvement. With respect to off-load planning, in the end-user account, alternative surface assault schedules and their representation (domain issues) are related alongside the reception of intelligence (worksystem issues) and are expressed in the specific terms of planning evacuations, such as 'tactical option' and 'gantt chart'. The pre-conception's account, in contrast, expresses only the former (surface assault schedules and their representation) and as level '3' plan objects and their view. The pre-conception's account also excludes such worksystem issues as the commander's lack of accurate information, and the stress, fatigue and training of various members of the command team. Thus, the end-user account may be said to be more explicit about worksystem issues and the physical characteristics of the domain, but such an emphasis is to be expected given the intended nature of the pre-conception.

Further comparison of the left and right hand columns of Tables 2.1. and 2.2., suggests that both the end-user account and the pre-conception supported account appear to describe the same events. With respect to the Vincennes incident, the event was one in which a neutral plane was shot down by a vessel that was simultaneously engaging a group of small boats. With respect to off-load planning, the hypothesised events were such that a number of related plans were developed incrementally and by selecting among options. The fact that the end-user account and model appear to refer to identical aspects of the real world suggests that the pre-conception has at least some face validity. Had re-formulating an incident in terms of the pre-conception resulted in mis-representation of the incident such that a different event seemed to be described, such a claim could not have been made.

2.2.2. Alternative Views of the Quality of Military Plans

In this section, the pre-conception-based view of the quality of specific types of military plan is compared and contrasted with an equivalent end-user's view. Plan quality, here, means the extent to which the plan produced is the plan that was desired. View of quality, here, means the conceptual dimensions of quality, rather than mere indices or levels of quality. For example, Brown et al.'s view of the quality of Fleet Employment Schedules comprises the dimensions: (i) 'fleet readiness' and 'morale'; (ii) 'level', 'necessity' and 'equability' of employment; and (iii) 'opportunity' for training and maintenance (1993). Naval Gunfire Support Plans, in contrast, seek to achieve appropriate 'coverage'. View of plan quality was selected as the basis of comparison, because end-user's views were relatively easy to elicit and minimal knowledge of particular types of military operations was required.

Two types of military plan are characterised - Off-Load Plans (that concern the off-load of men and equipment during amphibious operations) and SSGW Plans (plans for attack with Surface-to-Surface Guided Weapons). The selection of these types of plan was largely determined by pragmatic considerations, such as access to users. Following some background information about Off-Load and

SSGW plans, an end-user's views of their quality is presented. Then, the pre-conception-based views are presented and, finally, the end-user and pre-conception-based views are compared and contrasted. End-user's views of the quality of Off-Load Plans and SSGW Plans were elicited in a two-hour focussed interview. Questions posed included, 'What makes for a good Off-Load Plan/SSGW Plan?' and 'What's the difference between a good Off-Load Plan/SSGW Plan and a bad one?'. A more elaborate elicitation procedure could have been conducted, but time with users was limited.

2.2.2.1. Two Types of Military Plan

Off-Load Plans and SSGW Plans are similar in that they are both offensive and typically produced in parallel with plans for other aspects of the operation. The plans are different in that Off-Load Plans are typically produced a few days before the landing, and may govern immense tactical and logistic undertakings, possibly involving thousands of men and millions of tons of equipment. SSGW Plans, in contrast, are typically produced a matter of minutes or hours before an attack, and govern the use of a single weapons system.

Off-Load Plans. An amphibious operation may be considered as an attack against a potentially hostile shore, launched from the sea and involving air, sea and land forces. It includes the movement ashore of a landing force, embarked on transport ships and naval vessels, by means of amphibious vehicles, landing craft and helicopters. The landing force arrives ready for combat ashore, and at beaches and landing zones (rather than ports and airfields) (Evans, 1990). Critical for the landing are the Off-Load Plans (see Figure 2.4.). These plans specify who is to go ashore (left hand columns), and where, and when (right hand columns). They also specify who is to take the landing force ashore and how the landing force is to be tactically grouped (middle columns).

Figure 2.4: A Representative Off-Load Table

load	desired tactical order	no. men	from	by	timings depart	land	to
1	1 2 3 4	7 8 7 4	LSL1	LCA1	H-10	H	green
2	5 6 7	7 8 5	LSL1	LCA2	H-10	H	green
3	7 8 11	3 8 7	LSL1	LCA3	H-10	H	green
4	11 12	1 8	LSL1	LCA4	H-10	H	green
5 etc	14 15 etc	2 4 etc	LSL2 etc	LCA7 etc	H-15 etc	H+5 etc	red etc

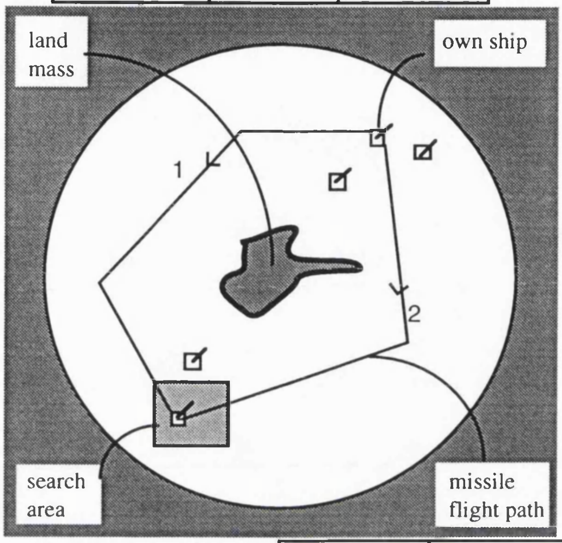
Key

LCA: landing craft assault
LSL: landing ship logistic
H-hour: time of landing for the first wave

SSGW Plans. Surface-to-Surface Guided Weapons are one component of a modern anti-surface warfare capability. Such systems play a crucial role in combatting enemy surface action (or battle) groups. The use of such weapons by a single ship is typically part of a larger, co-ordinated attack, which may involve many other ships, aircraft and submarines and which may last for a period of minutes of hours (Kiely, 1988; Till,1987). A representative SSGW Plan is shown in Figure 2.5. The table specifies the type of salvo and the number of missiles to be fired (upper rows). It also specifies each missile's terminal manoeuvres, where and how the missile is to search for its target, and the timing of the attack (lower rows). The diagram specifies the missile's flight path. The plan also includes target information (top row of the table) and depicts background shipping and land-masses. Given the definition of plans in section 1, target information may not be, strictly speaking, part of a plan. (It could be part of a picture, however.) To introduce SSGW Plans, however, we defer to the user.

Figure 2.5: A Representative SSGW Plan

Target Information	**range**	40nm
	bearing	220'
Salvo	**type**	all arrive at a set time
	number of missiles	2



		Missile 1	**Missile 2**
Launch	**canister**	starboard 1	port 2
Terminal Phase	**homing radar setting 1**	range and bearing search	range and bearing search
	homing radar setting 2	normal	normal
	terminal manoeuvre	pop up	sea-skimming
	re-attack	yes	yes
	fuse	'y' m/sec delay	'y' m/sec delay
Timings	**launch**	XX+2:00	XX+1:50
	on target	XX	XX

2.2.2.2. The End-User Views

In the case of Off-Load Plans, an interview with an end-user suggested that a high quality Off-Load Table results in a simple, rapid, and prioritous landing. A landing must be simple in the sense that the control of the ship-to-shore movement of air and sea traffic is greatly facilitated if dedicated air and sea lanes are specified. An Off-Load Plan must accommodate the possibility of such lanes. A landing must be rapid in order to maintain the element of surprise and to minimise the risks to the amphibious force when surprise is lost. A landing must be in order of priority, because the landing force must be prepared to fight when ashore. Prior to the landing, each element of the landing force is given a 'priority' number, which indicates the sequence in which various elements are to land. This sequence is tactically important and should be preserved during the landing.

In the case of SSGW Plans (see earlier), 'asking the user' suggested that a good plan for an SSGW attack should be based on good target information, surprise the enemy, achieve a high concentration of fire, and keep clear of background shipping. Good target information includes accurate and recent reports of the target's bearing, range, speed and heading. Surprise refers to the enemy's warning about the attack. If there is little warning, then the enemy will have little opportunity to defend itself, or plan its own pre-emptive strike. Concentration of fire refers to the chances of overwhelming the target's defences. Generally speaking, it is more difficult to counter a large number of missiles approaching from a single direction all at the same time, than a smaller number of missiles coming from a number of directions over a period of minutes. Clearance refers to the fact that the missiles' trajectory and profile should avoid friendly and neutral shipping, non-targetted enemy shipping, land masses and 'no-fire' zones.

2.2.2.3. The Pre-Conception-based Views

Views of plan quality were derived from the pre-conception by, first, associating a dimension of quality with each object or attribute from the domain of plans, and second, describing the dimension of 'appropriate content' in more detail by referring to objects and attributes in the domain of armed-conflict. This latter step was assisted by additional discussions with the user.

The pre-conception-based views of plan quality, then, are as follows. A high quality Off-Load Plan is well-scoped, well-represented, and specifies appropriate movement, lift, power and safety. Movement refers to the movement of the assault craft, helicopters and other means of getting ashore. Lift refers to the transportation such movement offers. Power refers to the power that the landing force projects ashore. Safety refers to the safety of the assault craft and elements of the landing force as they move.

A good SSGW Plan is well-scoped, well-represented and specifies appropriate movement, fire-power, acquisition, safety and surprise. Movement refers to the movement of the missile. Fire-power refers to the ability to bring one's weapons to bear on the target and acquisition refers to the information the missile acquires about the target. Safety refers to the safety of friendly and neutral forces (including the missile) and surprise refers to the enemy's surprise at being attacked.

2.2.2.4. Comparison

A comparison of the end-user and the pre-conception-based views of plan quality reveals some differences. First, and in common with the comparison of accounts of military incidents in section 2.2.1, end-user views are expressed in relatively concrete, operation-specific terms whereas the pre-conception-based views are expressed in more abstract and general terms. For example, an end-user view of good Off-Load plans was expressed in terms of 'speed, simplicity and in order of priority', and a good SSGW plan was expressed in terms of 'target information, surprise, concentration of fire and clearance'. The pre-conception-based view, in contrast, was expressed in a common language - that of scope, content, safety, movement etc. Second, and also in common with the accounts of military incidents, end-user views reflect the distinctive perspective of individuals responsible for producing or using plans, whereas the pre-conception-based views (by intent) maintain distinctions asserted Dowell & Long's conception (1989). For example, an end-user view of an SSGW plan included reference to 'target information' and tends to focus upon the nature of the plan's effect upon the conflict. The pre-conception-based view, in contrast, does not account for the first row in the SSGW plan since, strictly speaking, such information is not part of a plan - it concerns the *actual* state of armed-conflict, rather than its desired state. The pre-conception based view also explicitly recognises dimensions of plan quality, such as its view and scope, that may be conventional and presumed by an end-user, but highly relevant to HCI. Third, the pre-conception-based view also reflects the limitations inherent in its preliminary status. For example, the pre-conception-based view of an SSGW plan does not account for the third row of the plan, which concerns the seeker settings for the missile's homing radar. This row specifies a procedure for the missile to follow (turning on its radar, beginning to search etc.), but the pre-conception only identifies goals of military work, and not actions of military equipment.

Further comparison of end-user and pre-conception-based views suggest that both views appear to refer to the same aspects of the same artefacts, that is, documents which set goals for certain fighting forces, and which state goals for the appropriate range of entities, include the appropriate information and are represented the appropriate way. The pre-conception, then, may have at least some face validity.

This review concludes the pre-conception of the domain of C^2. In the following chapter, the pre-conception is used to support evaluation and specification.

Chapter 3.
Carry Forward in Late Evaluation

Summary
This chapter reports an initial attempt to realise carry forward in late *evaluation* in a manner characteristic of informal HCI Engineering. The attempt is compared and contrasted with the explicit Craft alternative and the adequacy with which the desired manner of carry forward is illustrated, is considered.

In the informal Engineering evaluation, the application representations involved are a Problem hierarchy and an expression of the general Problem of evaluating military planning systems. The Problem hierarchy is developed by synthesizing contrasts between C^2 and other types of work initially suggested during the development of the pre-conception. The expression of the planning system evaluation Problem is developed by specialising an expression of the more general Problem of evaluating human-computer interactions. The design work concerns a system for planning the off-load of men and equipment during amphibious military operations reconstructed in the laboratory for the purposes of this research - University College London's (UCL's) Off-Load Planning System (OPS). Two prototypes, OPS0.5 and OPS1, are implemented. The use of the application representations in evaluation begins with use of the Problem hierarchy to assist the technical interpretation of UCL's problem, which involves off-load planning, as an instance of the more general class of Problem, which involves military planning in general. This categorisation of UCL's problem increments the Problem hierarchy through the addition of a new instance and prompts the instantiation of the planning system evaluation Problem for off-load planning. A focus for the evaluation and indices of aspects of off-load planning effectiveness are selected. Finally, the desired effectiveness of OPS1 is set, its actual effectiveness obtained and the conformance between desired and actual effectiveness is assessed.

In the equivalent explicit Craft evaluation, an identical design context (implementations of OPS0.5 and OPS1) is assumed. The application representations involved are the views of planning system effectiveness used in previous evaluations, and views of effectiveness in general. The

use of the application representations begins with the perception of a similarity between OPS1 and one or more other military planning systems, and the incorporation of the associated views of effectiveness in the view of OPS1's effectiveness. However, the relationship between OPS1 and these other systems, and the selectivity of the views of effectiveness associated with these other systems, are poorly specified. As such, there is little reason to believe that simply copying any one view will be sufficient for the evaluation of OPS1. Consequently, elements of many previous views are selectively re-used, and a view of effectiveness in general interpreted, for the purposes. Assuming an identical focus and indices as for the informal Engineering evaluation, the desired effectiveness of OPS1 is set, its actual effectiveness obtained and the conformance between desired and actual effectiveness assessed.

Comparison between the informal Engineering and the explicit Craft evaluations suggests that, in this case, both evaluations addressed almost all aspects of effectiveness that would be expected of a late evaluation, and expressed effectiveness in a concise way. The main findings of the two evaluations were also equivalent. The informal Engineering evaluation, however, was different from the explicit Craft evaluation in terms of how the problem was conceived and how the completeness of the evaluation was reasoned about.

Consideration of how well this report illustrates carry forward in informal HCI Engineering, suggests that informal Engineering was partially realised in that an application representation (the Problem hierarchy) was incremented as a result of its use. Also, knowledge about classes of design Problem and instances of classes of design Problem was applied to support the development of an instance requirement and an instance specification. An informal class of Problem was formulated, and an instance of this class of Problem was addressed. Further, the additional means of reasoning about the completeness of design representations provided by informal Engineering was used. However, general (class) requirements and general (class) specifications were presumed rather than made explicit and the artefact was not conceived as an instance of a class of artefact - OPS1 is a bespoke implementation, which is presumably related to other implementations, but in unstated ways. Design work only comprised a transformation from artefact to instance specification, and from client requirement to instance requirement, followed by an assessment of the instance specification with respect to the instance requirement, and so opportunities for carry forward in addition to those available within explicit Craft were not, in actual fact, fully created or exploited. Also, carry forward was only partially realised in that an expression of the general planning system evaluation Problem was developed from the pre-conception and the HCI(e) conception, and instantiated for two particular planning system evaluation Problems at hand. However, re-use was not fully carried through. The view of SSGW planning performance is incomplete, since a view of the costs incurred by SSGW planning systems is not made explicit, and an SSGW worksystem is not, in actual fact, evaluated, only a view of its effectiveness is developed (which is an initial step).

Overall, judgements about the relative effectiveness in practice of the informal Engineering and explicit Craft evaluations rest upon: (i) the perceived value of the savings of time and effort achieved by reasoning about completeness with respect to classes of Problem, rather than interpreting other views for the purposes; and (ii) the perceived value of the additional investment required to develop the pre-conception, Problem hierarchy and Problem expressions, rather than document case histories and acquire general views of effectiveness. (It is assumed that the informal Engineering and explicit Craft evaluations are of approximately equal value in terms of their support for the development of subsequent user requirements. That is, they are equally insightful about planning effectiveness and the state of OPS1.) In this case, and in the author's opinion, the investment is judged to outweigh the savings. That is, the cost of developing the pre-conception etc. seems not to be justified by the ease of reasoning about completeness, at least in this instance and relative to the alternative option.

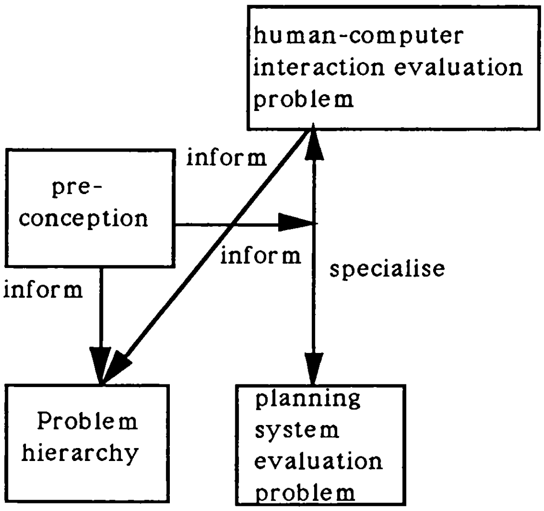
Subsequent attempts to realise carry forward in a manner characteristic of informal HCI Engineering, then, must illustrate carry forward more fully, ensure that a general (class) requirement and a general (class) specification are developed, and construct application representations that provide greater savings in development time and effort when designing instances.

3.1. Informal Engineering Late Evaluation

3.1.1. Development of Application Representations

In this attempt to realise carry forward in late evaluation in a manner characteristic of informal HCI Engineering, two application representations are developed - a Problem hierarchy and a view of the general evaluation Problem of military planning systems. The development of both application representations is supported by the pre-conception (see Figure 3.1).

Figure 3.1. Development of an Expression of the Planning System Evaluation Problem and a Problem Hierarchy



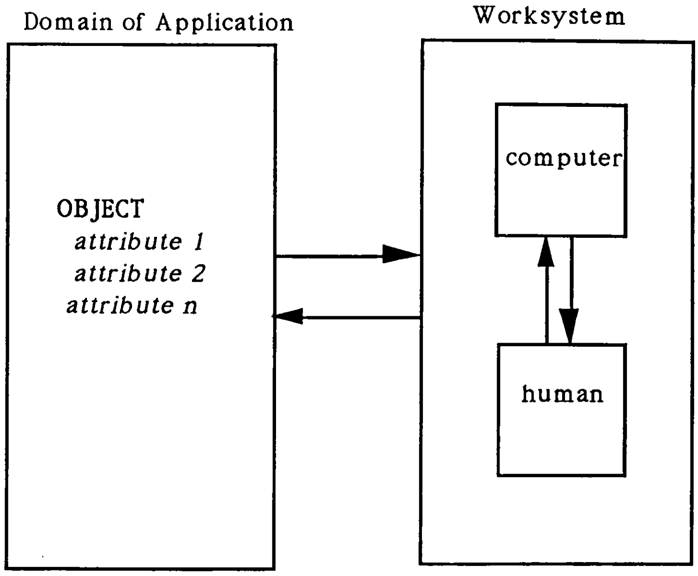
3.1.1.1. Development of a View of Planning System Effectiveness

In this Section, the general evaluation Problem, which comprises humans interacting with computers to perform work effectively, is systematically specialised using the pre-conception of C^2 to express a class of evaluation Problem, which comprises planners interacting with planning aids to produce plans effectively.

3.1.1.1.1. Humans Interacting with Computers and Performance

Dowell & Long's conception of HCI provides the opportunity for a class (superordinate) expression of the general evaluation Problem. This Problem comprises humans interacting with computers to perform work (see Figure 3.2) and their effectiveness. Specialisation of this expression for subordinate classes of evaluation Problem may express the distinctive features of more specific concerns while retaining the characteristics of the class (see later).

Figure 3.2. A Human Interacting with a Computer to Perform Work (after Dowell & Long, 1989)



In the conception, an application domain (of a worksystem) is where work originates, is performed and has its consequences (see Section 1.2.3). It comprises one or more objects constituted of attributes, which take on a variety of attribute states. Task goals express a requirement for change in the state of these attributes, and goals are allocated to worksystems by organisations. A domain is distinct from, and delimits, a worksystem. A worksystem comprises at least two separate, but interacting sub-systems - a sub-system of human behaviours interacting with a sub-system of computer behaviours. These human and computer behaviours are supported by mutually exclusive human and computer structures and are executed in order to perform tasks effectively.

For Dowell & Long, effectiveness is expressed through the concept of performance, that is, how well a worksystem achieves its goals ('task quality'), and the costs that are incurred in so doing ('system costs'). Costs are incurred both by the human and the computer and are structural and behavioural[1].

---

[1] For convenience, worksystem performance may be summarised thus:

$$P_{work} \equiv Q_{work} \times K_{work} \quad (1)$$

Where P_{work} represents Worksystem Performance, Q_{work} represents Task Quality, and K_{work} represents Worksystem Costs.

Thus, the superordinate class of Evaluation Problem may be expressed as:

for $U^B \times C^B$, find where P_R =/= P_A

where P_R represents required performance and P_A represents actual performance.

3.1.1.1.2. Planners Interacting with Planning Aids and Planning Performance

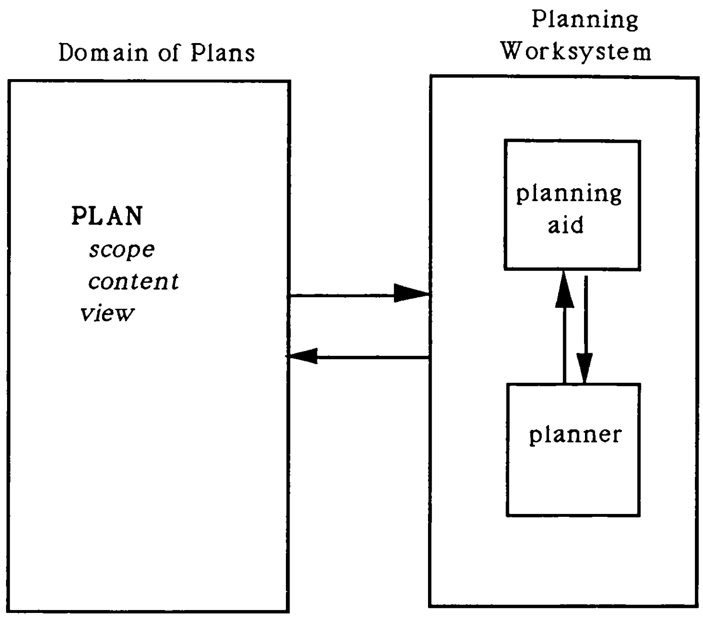
Plans (documents), their potential for change and the realisation of this potential constitutes a class of work - planning work. Human planners interacting with computer-based planning aids constitute a class of worksystem - planning worksystems (see Figure 3.3). In turn, this class belongs to the more general class of humans interacting with computers to perform work effectively (see previous section).

Worksystem Costs may be further defined:

$$K_{Work} \equiv KU^S_{Work} \times KU^B_{Work} \times KC^S_{Work} \times KC^B_{Work} \quad (2)$$

where *U* represents the User, *C* represents the Computer, *S* represents structural costs and *B* behavioural costs.

Figure 3.3. A Planner Interacting with a Planning Aid to Produce Plans



To summarise the relevant part of the pre-conception (see section 2.1.2.1), the domain of plans comprises a single type of object - plans. A plan, here, is a representation of goals and/or procedures for the work of controlling operations and has at least three attributes: (i)*scope* - defines and delimits content; (ii) *content* - what is specified or 'meant by' the plan; and (iii) *view* - the type of language or representational scheme through which content is communicated. Given this conception of the planning domain, planning tasks may be expressed as required changes in scope, content, view. For example, a planning task may require the production of a table and illustrative diagram (view) showing the movement (content) of 152 Squadron for tomorrow's attack (scope). Further, it is possible to categorise the interactive planning behaviours of planning worksystems in terms of these attributes - one category for each task goal pursued. That is, planning worksystems scope plans, consider their content, and represent content in different views. A planning worksystem's structures may be similarly specified. That is, planning worksystems possess structures for scoping plans, considering their content and representing plans differently. Thus, dimensions of plan quality and further divisions of worksystem cost may be derived from the pre-conception - one dimension of quality and one further division of cost for each task goal pursued. That is, a good plan is well-scoped, has desirable content and may be viewed in various ways. Human structural costs may be further divided into human structural costs incurred scoping the plan, human structural costs incurred considering the content of the plan, and human structural costs incurred producing different views of the plan[2].

Thus, the sub-class of evaluation Problem comprising human planners, planning aids and plans, then, may be expressed as:

for $U^{B}_{Planning(x)} \times C^{B}_{Planning(x)}$, find where $P\mathrm{R}_{Planning(x)} =/= P\mathrm{A}_{Planning(x)}$

where $P\mathrm{R}_{Planning(x)}$ represents required planning performance and $P\mathrm{A}_{Planning(x)}$ represents actual planning performance.

3.1.1.2. Development of a Problem Hierarchy

During the development and review of the pre-conception, the domain of C^2 was distinguished from a number of related kinds of work (see Chapter

---

[2] For convenience the Planning for some generic planning task X is defined:

$$Planning(x) \equiv C(x) \times S(x) \times V(x) \qquad (3)$$

Where C represents Content, S represents Scope, and V represents View.

From definitions (1) and (3) the following equation may be obtained:

$$P_{Planning(x)} = Q_{(C(x),\ S(x),\ V(x))} \times K_{(C(x),\ S(x),\ V(x))} \qquad (4)$$

In general for any aspect of performance, a, with different aspects of work, g and h, we write the following:

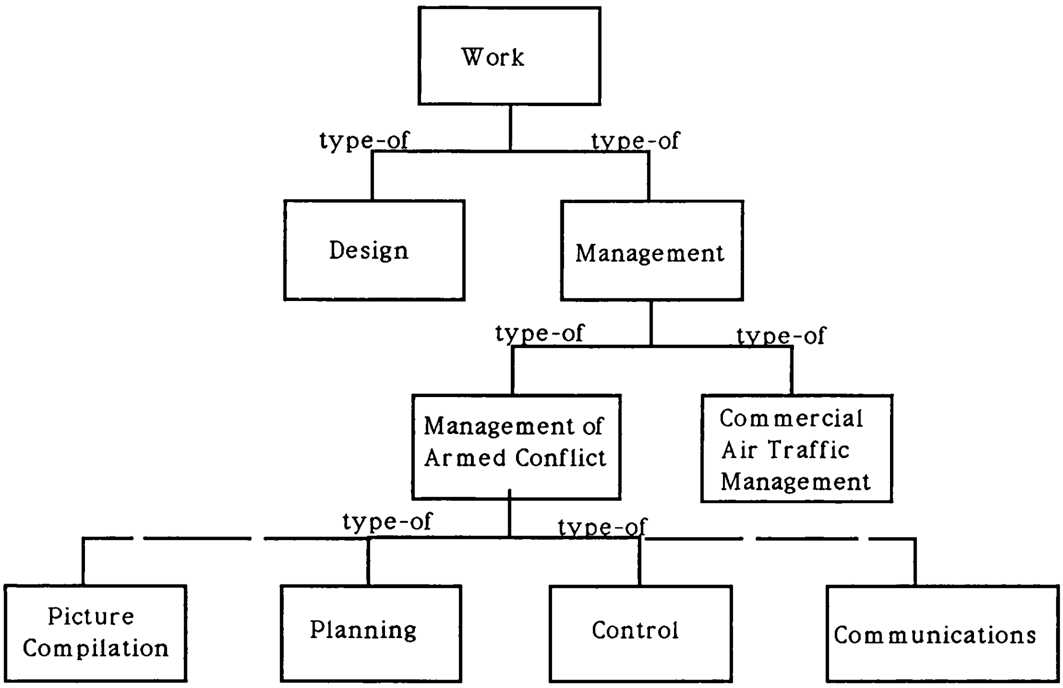
$$\alpha_{(\gamma,\ \eta)} \equiv \alpha_{\gamma} \times \alpha_{\eta} \qquad (5)$$

Thus from (5) the following holds:

$$Q_{(C(x),\ S(x),\ V(x))} = Q_{C(x)} \times Q_{S(x)} \times Q_{V(x)} \qquad (6)$$

2). These relations are now expressed in terms of member<-->class relations and represented within a Problem hierarchy (see Figure 3.4).

Figure 3.4: Problem Hierarchy



3.1.2. Development of an Off-Load Planning Worksystem for University College London

A hypothetical client requirement expressed by University College London (UCL) was presumed and remained implicit. A HyperCard prototype was implemented - Off-Load Planning System (OPS) version 0.5 (see Figure 3.5). The prototype simulated a computer system which, together with its student user, generates and assesses a simplified plan for the off-load of men and equipment during hypothetical amphibious operations. Implementation was assisted by demonstrations of other military and non-military planning systems. These demonstrations suggested that a prototype with more advanced facilities, such as a function that detected clerical errors in the plan, and a variety of types of explanation display also had the potential to satisfy the presumed client requirement. This more advanced prototype was labelled OPS1.0 (see Figure 3.6).

File Edit Go Tools Objects Font Style

Off-Load Plan

Load No.	Contents	No. People	No. Vehicles	Desired Tactical Order	From	To	Timing Depart	Timing Land	Means
1	40CDY/001	8	0	4	SHIP 1	BEACH 1	06:30	06:50	LCP 1
	40CDY/002	8	0	5	SHIP 1	BEACH 1	06:30	06:50	LCP 1
	40CDY/003	8	0	6	SHIP 1	BEACH 1	06:30	06:50	LCP 1
2	40CDY/004	7	0	7	SHIP 1	BEACH 1	06:30	06:50	LCP 2
	40CDY/005	4	0	8	SHIP 1	BEACH 1	06:30	06:50	LCP 2
	40CDY/006	7	0	9	SHIP 1	BEACH 1	06:30	06:50	LCP 2
	40CDY/007	4	0	10	SHIP 1	BEACH 1	06:30	06:50	LCP 2

Next Load - Pending

Load No.	Contents	No. People	No. Vehicles	Desired Tactical Order	From	To	Timing Depart	Timing Land	Means
3	40CDY/008	7	0	11	SHIP 1	BEACH 2	06:30	06:50	LCP 3
	40CDY/009	8	0	12	SHIP 1	BEACH 2	06:30	06:50	LCP 3
	40CDY/010	7	0	13	SHIP 1	BEACH 2	06:30	06:50	LCP 3
	40CDY/011	4	0	14	SHIP 1	BEACH 2	06:30	06:50	LCP 3

Approve

Next Load - Options

Load No. 3

Option No.	Assessment	Contents (order no.)	Means
1.	100	11,12,13,	LCU 3
2.	80	11,12,13,14	LCU 3
3.	15	11,12,13,	LCU 1
4.	50	16,17,18,19,	LCVP 1
5.	60	11,12,13,15	LCU 3

Next Load - Assessments

Option 2

Criteria	Score
Power :	30
Lift :	20
Safety :	0
Cohesion :	20
Fatigue :	10
Total :	80

Figure 3.5: OPSO.5

File Edit Go Tools Objects Font Style

Off-Load Plan

Load No.	Contents	No. People	No. Vehicles	Desired Tactical Order	From	To	Timing Depart	Timing Land	Means
1	40COY/001	8	0	4	SHIP 1	BEACH 1	06:30	06:50	LCP 1
	40COY/002	8	0	5	SHIP 1	BEACH 1	06:30	06:50	LCP 1
	40COY/003	8	0	6	SHIP 1	BEACH 1	06:30	06:50	LCP 1
2	40COY/004	7	0	7	SHIP 1	BEACH 1	06:30	06:50	LCP 2
	40COY/005	4	0	8	SHIP 1	BEACH 1	06:30	06:50	LCP 2
	40COY/006	7	0	9	SHIP 1	BEACH 1	06:30	06:50	LCP 2
	40COY/007	4	0	10	SHIP 1	BEACH 1	06:30	06:50	LCP 2

Next Load - Pending

Load No.	Contents	No. People	No. Vehicles	Desired Tactical Order	From	To	Timing Depart	Timing Land	Means
3	40COY/008	7	0	11	SHIP 1	BEACH 2	06:30	06:50	LCP 10
	40COY/009	8	0	12	SHIP 1	BEACH 2	06:30	06:50	LCP 10
	40COY/010	7	0	13	SHIP 1	BEACH 2	06:30	06:50	LCP 10
	40COY/011	8	0	14	SHIP 1	BEACH 2	06:30	06:50	LCP 10

Clerical Check — Show Option: 1 2 3 4 5 — Approve

Next Load - Options

Load No. 3

Option No.	Assessment	Contents (order no.)	Means	Journey Time
1.	OK	11,12,13,	LCP 3	20 mins
2.	Overloaded	11,12,13,14	LCP 3	20 mins
3.	Overloaded	11,12,13,	LCP 4	20 mins
4.	Not Desired Order	16,17,18,19,	LCP 1	35 mins
5.	Not Desired Order	15,16,17,18,	LCP 3	20 mins

Next Load - Assessments

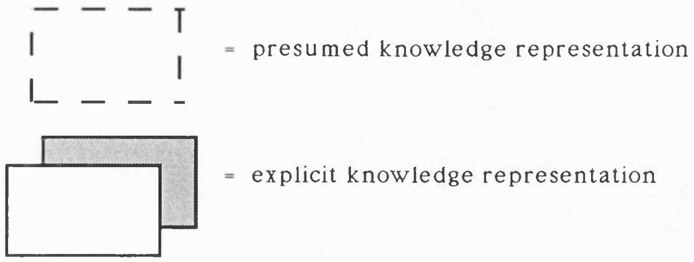
Option No.	Total	Power	Lift	Safety	Cohesion	Fatigue
1.	100	30	20	20	20	10
2.	80	30	20	0	20	10
3.	100	30	20	20	20	10
4.	50	20	10	20	0	0
5.	70	10	20	20	10	10

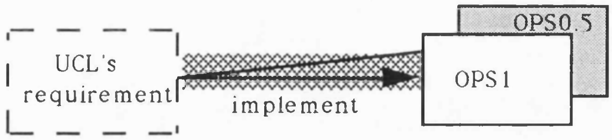
Assess — Compare — All Scores — Clear Away

Figure 3.6: OPS1

Figure 3.7: Design Work: Implementation of OPS0.5 and OPS1

KEY:





3.1.3. Use and Incrementation of Application Representations in Evaluation

The Problem hierarchy was used to assist the technical interpretation of UCL's requirement as an instance of the presumed requirement for military planning systems (see Figure 3.8). On initial inspection, and given the alternatives, UCL's requirement appeared to implicate a novel sub-class of evaluation Problem - off-load planning system evaluation - which exhibited the characteristics of the military planning system evaluation Problem in general. (UCL is presumed to be one of many organisations that may express requirements for off-load planning systems). The categorisation of off-load planning as a sub-class of military planning extended the Problem hierarchy through the addition of the new sub-class (see Figure 3.9) and prompted the instantiation of the planning system evaluation Problem for off-load planning at UCL. Having formulated an expression of the evaluation Problem instance at hand, the completeness of the evaluation was considered. Next, the procedure by which statements of performance were obtained was devised. Then, the required performance (Pr) of OPS1 was set and its actual performance (Pa) obtained. Finally, the conformance of required and actual performance of OPS1 is considered. To focus on effectiveness in practice and different manners of carry forward, some conventionally reported aspects of evaluation, such as the rationale for the choice of metrics and data analysis, operational scenario, etc., are omitted (see Section 1.2.4).

Figure 3.8: Use and Incrementation of Problem Hierarchy and Evaluation Problem Expression: the case of OPS1

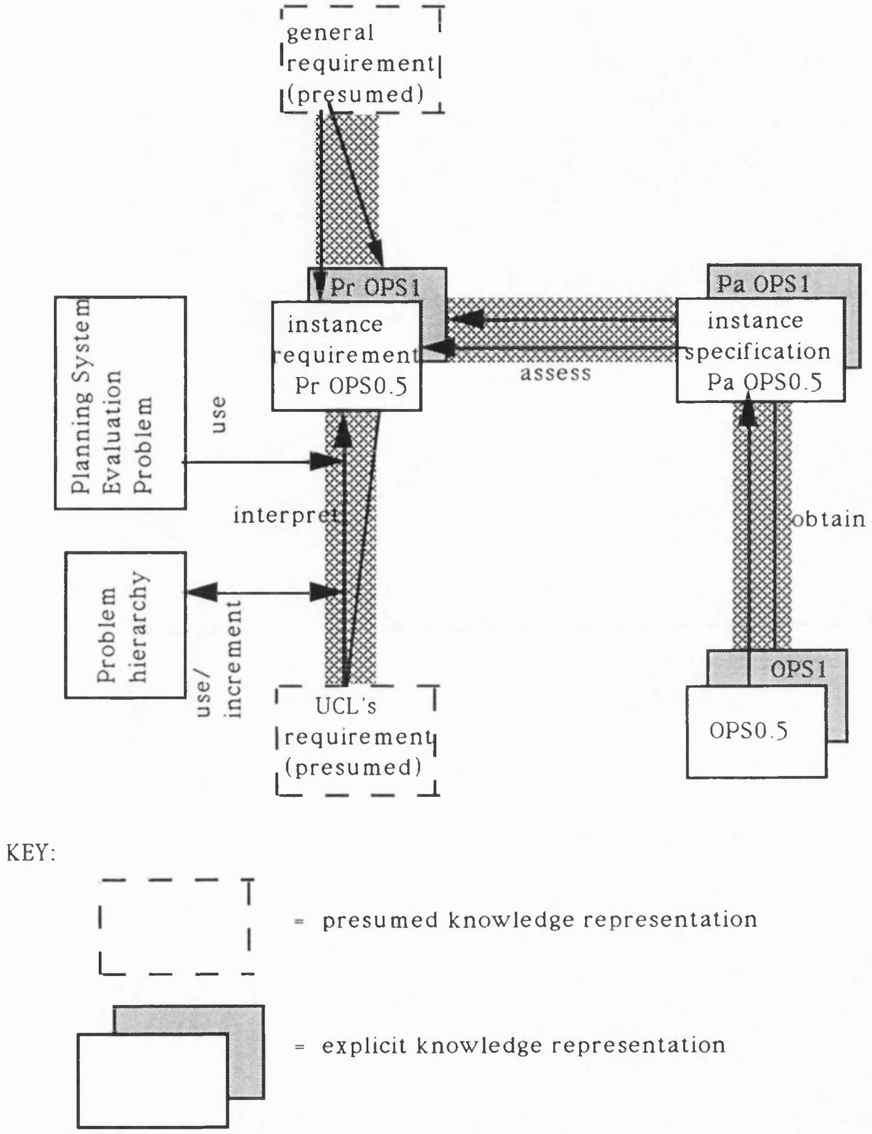
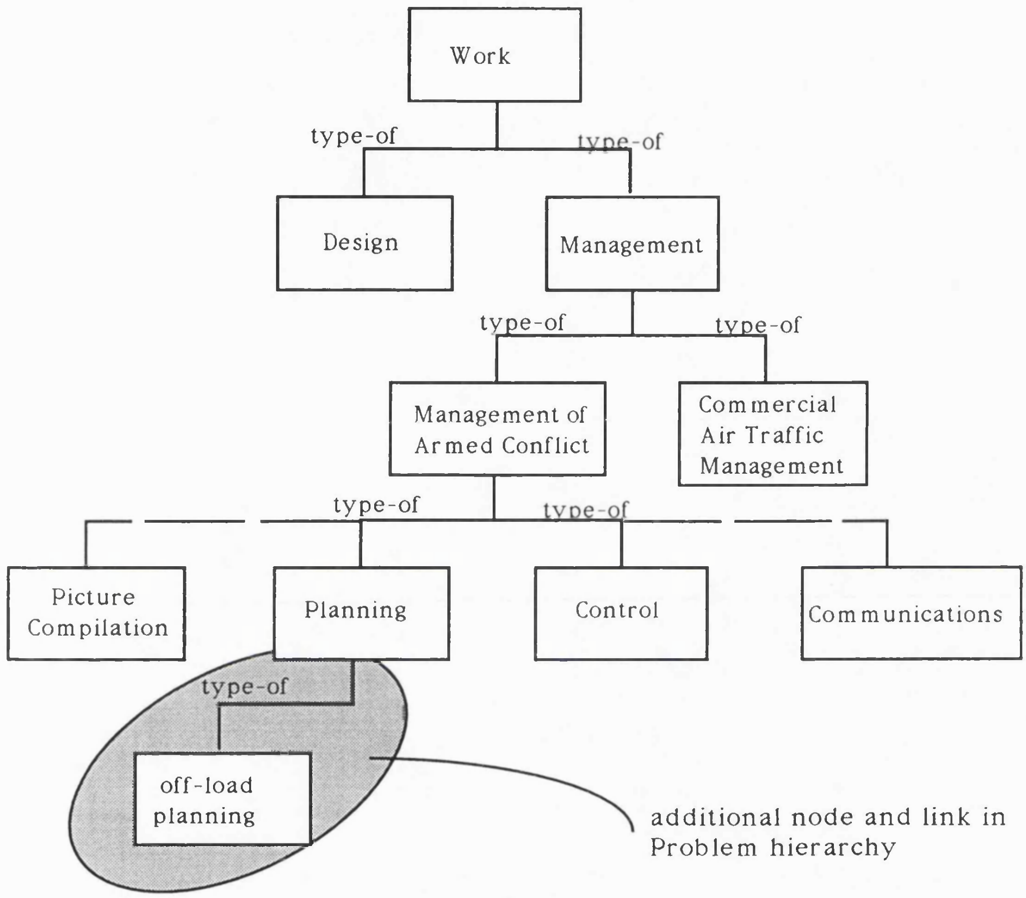


Figure 3.9: Incremented Problem Hierarchy

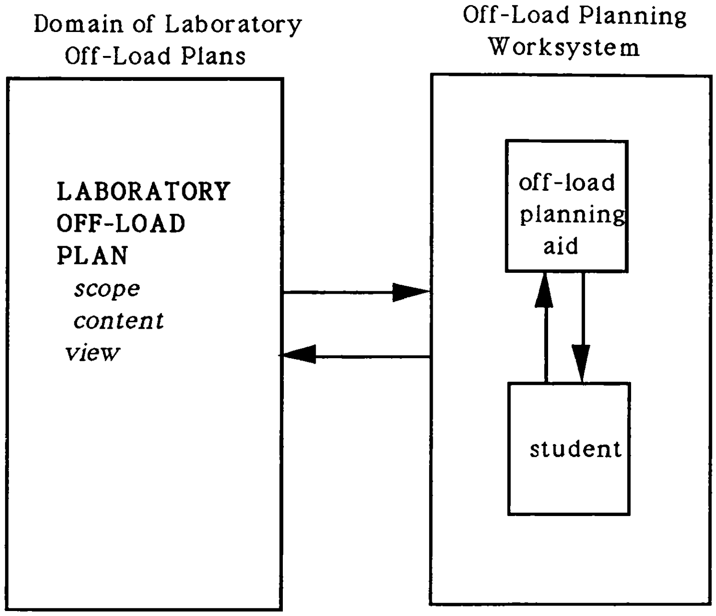


The instantiation of the general military planning system evaluation Problem for off-load planning with OPS1 at UCL follows.

3.1.3.1. Students Interacting With an Off-Load Planning Worksystem (OPS1) at University College London and Off-Load Planning Performance

The reconstructed and simplified plans for off-loading (notional) men and equipment during amphibious (military) operations and their production for purposes of research at University College London (UCL) is an instance of planning work, namely, laboratory off-load planning work. Student subjects interacting with a simulated off-load planning worksystem at UCL constitutes a specialisation of a planning worksystem (in general) which is, in turn, a specialisation of a worksystem (yet more generally)(see Figure 3.10)[3].

Figure 3.10. A Student Interacting with an Off-Load Planning Aid at University College London to Produce Laboratory Off-Load Plans



The laboratory plans produced are plans for off-loading men and equipment from landing ships during amphibious (military) operations (see upper window of Figure 3.6). An off-load plan specifies who is to go ashore, where and when, and what is to transport them (see section 2.2.2.1). For laboratory off-load plans: *scope* refers to the assault craft and elements of the landing force for which goals have been set (as opposed to those for which goals have not been set) and the period of time to which those goals are to apply; *content* concerns the goals that have been set for movement and lift of assault craft, the fatigue, cohesion and power of the landing force and the safety of both assault craft and landing force; and *view* refers to the display of the plan as a table.

---

[3]Off Load Planning is an specialisation of Planning, so for example from (3) and (6) we obtain the following:

$$Q_{Planning(OffLoadLab)} = Q_{S(OffLoadLab)} \times Q_{C(OffLoadLab} \times Q_{V(OffLoadLab)}$$

The laboratory off-load planning task requires an initial plan to be modified. This task and planning performance is considered in more detail when desired performance is set (see later).

The laboratory Off-load Planning Worksystem (OPS1), comprises a HyperCard planning aid and student subjects, whose sole justification is to support this research. Consequently, some of the planning aid's behaviours are simulated, rather than implemented, and most are considerably simpler than those of an actual off-load planning aid. Further, since it only exhibits *some* of the behaviours characteristic of planning systems as a class, it may be regarded as an (intentionally) incomplete system. The system is reconstructed in the sense that it was developed following analysis of an actual demonstrator system (see Acknowledgement)

In OPS1, content is generated (loads are added to the off-load table) one line at a time, starting from the beginning of the landing. A student subject 'approves' one of the load options offered and the planning aid adds the text in the 'Next Load - Pending' window to the bottom of the off-load plan (see Figure 3.6). As part of *considering content*, the computer generates and displays summaries of five alternative options for the next load in the 'Next Load - Options' window. If a student subject clicks on an option of interest, the details of that option are displayed in the 'Next Load - Pending' window. OPS1 also assesses the chosen option in a number of ways and against a number of criteria. A student subject may choose the type of assessment they wish to see by pressing the appropriate button in the 'Next Load - Assessment' window. To ensure that student subjects, who are somewhat naive about off-load planning, considered content actively, rather than simply approving whatever load the computer suggested, 'bugs' were deliberately introduced into OPS1's (simulated) load assessment algorithm. For example, a load option that is, in actual fact, rather fatiguing and unsafe because the assault craft is overloaded, may be erroneously rated highly by OPS1. A student subject has to double-check OPS1's reasoning and adjust the load options' scores to compensate for the bugs. A formatted Notepad was provided to help student subjects double-check systematically. Student subjects actively considered the *scope* of the laboratory off-load plan by, similarly, locating and correcting deliberately introduced bugs. In this case, the bugs were in OPS1's (simulated) assault craft and landing force selection algorithm, and student subjects were assisted by a 'clerical check' facility. For example, a load option may involve a landing craft that is not, in fact, participating in the operation. In OPS1, only a single view of an off-load plan is possible - as a table - so viewing behaviour may be regarded, for most practical purposes, as impossible. Consequently, OPS1 must be regarded as an (intentionally) incomplete planning system.

OPS1's structures (the student subjects' mental representations and processes and the planning aid's stacks and scripts) may be similarly distinguished as structures for scoping off-load plans and considering their content. For example, the knowledge that the subject acquired during training about OPS1, amphibious operations and off-load planning, constitute human structures for considering content, and the HyperCard stacks and scripts, that simulate an option generation algorithm, constitute a computer structure for considering content.

Having conceived the simplified off-load planning tasks, OPS1 and students as a specialisation of plans, planners and planning aids, the evaluation Problem in the instance may be expressed as:

for $U^{B}_{Planning(OffLoadLab)} \times C^{B}_{Planning(OffLoadLab)}$, find where $P\mathrm{R}_{Planning(OffLoadLab)}$ =/= $P\mathrm{A}_{Planning(OffLoadLab)}$

where $P\mathrm{R}_{Planning(OffLoadLab)}$ represents required off-load planning performance and $P\mathrm{A}_{Planning(OffLoadLab)}$ represents actual off-load planning performance.

This section completes the expression of the evaluation Problem instance of off-load planning with OPS1 at UCL.

3.1.3.2. Focus of the Evaluation

Because of resource limitations, only a selective evaluation was conducted. The focus of the evaluation was determined by the need to illustrate carry forward characteristic of informal HCI Engineering. The aspects of performance selected are: two aspects of plan quality - scope and content (plan content is taken to concern lift only); user and computer behavioural costs associated with considering content; and overall user and computer structural and behavioural costs. (Overall, here, means over all behaviours). Other aspects of off-load planning performance, such as how well the plan is represented and the cost associated with representing the plan and scoping the plan, are not addressed in the evaluation. The focus of the evaluation upon these aspects of performance reflects the limitations of the planning aid. Since only some aspects of the performance of planning systems (as a class) are addressed in this instance, the evaluation is incomplete (or, rather, highly selective). The selectivity, however, is by intent, explicit and well-specified.

3.1.3.3. Procedure for Obtaining Statements of Performance

In the evaluation, the actual performance of OPS1 was obtained by observing five student subjects learning to use and subsequently using the system. Training required student subjects to read background material, watch demonstrations, explore OPS1 and complete multiple-choice tests assessing what they had learnt. Following training, student subjects were asked to produce two practice lines/loads for an off-load plan. Then, each student subject attempted to produce five more lines/loads of the off-load plan within 50 minutes. Student subjects were unobtrusively observed and informally debriefed.

3.1.3.4. Metrics

A number of performance indices were used in the evaluation (see Table 3.1). Indices were selected for ease of data collection and for their adequacy in supporting the illustration. They have no importance here *per se*. The *quality of the scope* of the laboratory off-load plan was indicated by: (i) errors concerning the landing force or assault craft, that is, confusions and innaccuracies in Columns 2-4 inclusive and Column 10 of the off-load plan. (This measure particularly refers to *object_Scope* (see section 2.1.2.1)); and (ii) the mean number of lines/loads of the plan that had been completed by the deadline, and expressed as a percentage. (This measure particularly refers to *time_Scope* (see Section 2.1.2.1.), since the time period to which the plan applied increased as additional lines of the plan were added. The *quality of the content* of the laboratory off-load plan was indicated by the planned rate of lift, that is, the rate at which men and equipment were due to be off-loaded, in terms of men per hour.

Table 3.1. Performance Indices for the Evaluation of Laboratory Off-Load Planning Worksystems

Concept	Index
Quality of Laboratory Off-Load Plans $rQ_{Planning(OffLoadLab)}$	
Content $Q_{C_{Lift}(OffLoadLab)}$	mean planned rate of lift (men/hr.)
Scope $Q_{S(OffLoadLab)}$	mean no. errors in Columns 2,3,4, & 10 of plan mean percentage of plan completed by the deadline (%)
Costs incurred by Student Subjects $rKU_{Planning(OffLoadLab)}$	
Overall Structural $KU^{S}_{Planning(OffLoadLab)}$	mean duration of exploration mean no. of correct answers on test
Overall Behavioural $KU^{B}_{Planning(OffLoadLab)}$	mean workload rating
Behavioural (associated with Considering Content) $KU^{B}_{C(OffLoadLab)}$	mean no. of notepad entries for the last 2 lines of plan
Costs incurred by Laboratory Off-Load Planning Aids $rKC_{Planning(OffLoadLab)}$	
Structural $KC^{S}_{Planning(OffLoadLab)}$	lines of code (excluding initialisation, and data collection) no. interface objects no. handlers that call handlers on other stacks
Behavioural $KC^{B}_{Planning(OffLoadLab)}$	mean time to produce a plan
Behavioural (associated with Considering Content) $KC^{B}_{C(OffLoadLab)}$	estimated run time (standard interaction)

Overall user behavioural costs were indicated by the mean workload rating on a scale from 1 to 5. (It is assumed that student subjects can assess the rate at which they are incurring costs.) *User behavioural costs* associated with considering content were indicated by the mean number of entries in the 'Notepad'. (It is assumed that every time a student subject considers content, they make an entry in the Notepad).

Overall behavioural computer costs were indicated by the time taken to produce a five line plan. (It is assumed that the computer is constantly exhibiting behaviour, even if only maintaining a display, and that costs incurred are proportional to the length of time the structures required to support this behaviour are activated.) *Computer behavioural costs associated with considering content* were indicated by the estimated run time of the show-option and assessment scripts. (It is assumed that student subjects will wish to examine in detail, and view the computer's assessment of every load which they have considered in the Notepad.)

Overall user structural costs were indicated by the mean number of correct answers achieved on the multiple-choice tests during training and the average length of time student subjects spent exploring the device. (It is assumed that student subjects knew nothing about amphibious operations, off-load planning and the device prior to the study and that structural costs are incurred at a constant rate during exploration.)

Overall computer structural costs were indicated by the lines of code in the HyperCard scripts, the number of interface objects and the separability of different parts of the program (specifically, the number of handlers that call handlers located on other stacks). (It is assumed that smaller, more modular programs are easier for programmers to read and to write.)

3.1.3.5.Instance Requirement: Required Performance of OPS1

To simplify the illustration, the required performance of OPS1 is simply asserted in this section, either in absolute terms or relative to the actual performance of a previous version of OPS1 - OPS0.5[4]. First, a seven line

---

[4]A complete report of this evaluation, then, would also have included the implicit interpretation of OPS0.5 as an instance of a military planning system, obtaining the actual performance of OPS0.5, and the assessment of conformance between the required and actual performance of OPS0.5 that enabled the formulation of the required performance of OPS1 (see Figure 3.8). To simplify the illustration, however, this activity, and the knowledge

plan must be available 50 minutes after the start of the task. (These seven lines specify the first hour of the landing on one particular beach.) Second, the plan produced should specify lift at a rate of between 255 and 275 men/hr (presumed to be necessary to achieve the numerical superiority required. With OPS0.5, student subjects achieved a lift of 278 men/hr (see Table 3.2)). Third, the object_Scope of the plan must be better than that obtained with OPS0.5, that is, better than 1.8 clerical errors. Fourth, overall structural and behavioural user costs must be less than those incurred with OPS0.5, that is, not more than 35mins 39 secs to explore the device, at least 15.9 correct answers on the test and a mean workload rating of not more than 3.3, respectively. Fifth, user behavioural costs associated with considering content should be low. Specifically, the mean number of entries in the Notepad should be around six. (Given the bugs introduced, six is the minimum number of entries required to double-check OPS1 explicitly). Sixth, provided overall computer behavioural costs are less than those incurred by OPS0.5, that is, the task is completed in less than 42mins 54 secs, and other performance criteria are satisfied, overall structural computer costs may be slightly more than those incurred by OPS0.5, (a more effective interaction may require a larger program). Seventh, and finally, computer behavioural costs associated with considering content should be low. Specifically, it must be estimated to be less than 2.5 secs for a standard interaction, that is, one in which a student subject examines in detail each alternative load and views the computer's assessment of each load for which there is a Notepad entry. If time for a standard interaction is higher, then overall user costs are adversely affected. (Students become frustrated with the computer's slow response time, and complain that such delays interrupt their train of thought).

---

that supported it, has been omitted from the main body of the case history, since it is considered 'enabling' (see Section 1.2.4)

Table 3.2. Actual Performance of OPS0.5

Concept	Index
Quality of Laboratory Off-Load Plans $rQ_{Planning(OffLoadLab)}$	
Content $Q_{C_{Lift}(OffLoadLab)}$	mean proportion of plan available by the deadline: 100%
Scope $Q_{S(OffLoadLab)}$	mean planned rate of arrival: 278 men/hr. mean errors in Columns 2,3,4 & 10: 1.8
Costs incurred by Student Subjects $rKU_{Planning(OffLoadLab)}$	
Overall Structural $KU^{S}_{Planning(OffLoadLab)}$	mean duration of exploration: 35 mins. 39 secs. mean correct answers on test: 15.9
Overall Behavioural $KU^{B}_{Planning(OffLoadLab)}$	mean workload rating: 3.3
Behavioural (associated with Considering Content) $KU^{B}_{C(OffLoadLab)}$	mean no. of notepad entries for the last two lines of the plan: 11.2
Costs incurred by Laboratory Off-Load Planning Aids $rKC_{Planning(OffLoadLab)}$	
Structural $KC^{S}_{Planning(OffLoadLab)}$	167 lines of code 22 interface objects 2 handlers that call handlers on other stacks
Behavioural $KC^{B}_{Planning(OffLoadLab)}$	mean time to produce a plan: 42 mins. 54 secs
Behavioural (associated with Considering Content) $KC^{B}_{C(OffLoadLab)}$	2.5 secs estimated run time (standard interaction)

Table 3.3. Required Performance of OPS1

Concept	Index
Quality of Laboratory Off-Load Plans $rQ_{Planning(OffLoadLab)}$	
Content $Q_{C_{Lift}(OffLoadLab)}$	mean proportion of plan available by the deadline: 100%
Scope $Q_{S(OffLoadLab)}$	mean planned rate of arrival: 255-275 men/hr. not more than 1.8 errors in Columns 2,3,4 & 10
Costs incurred by Student Subjects $rKU_{Planning(OffLoadLab)}$	
Overall Structural $KU^{S}_{Planning(OffLoadLab)}$	mean duration of exploration: not more than 35 mins. 39 secs mean correct answers on test: at least 15.9
Overall Behavioural $KU^{B}_{Planning(OffLoadLab)}$	mean workload rating; not more than 3.3
Behavioural (associated with Considering Content) $KU^{B}_{C(OffLoadLab)}$	mean no. of notepad entries for the last two lines of the plan: between 4 and 8
Costs incurred by Laboratory Off-Load Planning Aids $rKC_{Planning(OffLoadLab)}$	
Structural $KC^{S}_{Planning(OffLoadLab)}$	lines of code: may be slightly more than 167 interface objects: may be slightly more than 22 handlers: 2 handlers may call handlers on other stacks
Behavioural $KC^{B}_{Planning(OffLoadLab)}$	mean time to produce a plan: not more than 42 mins. 54 secs;
Behavioural (associated with Considering Content) $KC^{B}_{C(OffLoadLab)}$	estimated run time (standard interaction): not more than 2.5 secs

3.1.3.6. Instance Specification: Actual Performance of OPS1

The actual performance of OPS1 was obtained by the procedure outlined in Section 3.1.3.3. (see Table 3.4.)

Table 3.4. Actual Performance of OPS1

Concept	Index
Quality of Laboratory Off-Load Plans $rQ_{Planning(OffLoadLab)}$	
Content $Q_{C_{Lift}(OffLoadLab)}$	mean proportion of plan completed by the deadline: 91.5%
Scope $Q_{S(OffLoadLab)}$	mean planned rate of lift: 267 men/hr.
	mean no. errors in Columns 2,3,4, & 10: 0.4
Costs incurred by Student Subjects $rKU_{Planning(OffLoadLab)}$	
Overall Structural $KU^{S}_{Planning(OffLoadLab)}$	mean duration of exploration: 32 mins. 31 secs. mean no. of correct answers on test: 15.9
Overall Behavioural $KU^{B}_{Planning(OffLoadLab)}$	mean workload rating : 3.0
Behavioural (associated with Considering Content) $KU^{B}_{C(OffLoadLab)}$	mean no. of notepad entries for the last 2 lines of plan: 7.7
Costs incurred by Laboratory Off-Load Planning Aids $rKC_{Planning(OffLoadLab)}$	
Structural $KC^{S}_{Planning(OffLoadLab)}$	199 lines of code; 40 interface objects; 1 handler calling handlers on other stacks;
Behavioural $KC^{B}_{Planning(OffLoadLab)}$	mean time to produce a plan: 40 mins. 6 secs.
Behavioural (associated with Considering Content) $KC^{B}_{C(OffLoadLab)}$	estimated run time (standard interaction): 1.5 secs.

3.1.3.7. The Conformance Between Actual and Required Performance of OPS1

This assessment of the conformance between OPS1's required (see Table 3.3) and actual (see Table 3.4) performance is summarised in Table 3.5.

Table 3.5. The Conformance between Required and Actual Performance of OPS1

Concept	Index
Quality of Laboratory Off-Load Plans $rQ_{Planning(OffLoadLab)}$	
Content $Q_{C_{Lift}(OffLoadLab)}$	Unacceptable (91.5% available by the deadline is less than 100%)
Scope $Q_{S(OffLoadLab)}$	Acceptable (a planned rate of lift 267 men/hr. is between 255 and 275 men/hr.) Acceptable (0.4 errors in Columns 2,3,4, & 10 is less than 1.8)
Costs incurred by Student Subjects $rKU_{Planning(OffLoadLab)}$	
Overall Structural $KU^{S}_{Planning(OffLoadLab)}$	Acceptable (32 mins. 31 secs. exploring, is less than 35 mins. 39 secs.) Acceptable (15.9 correct answers on test, is not less than 15.9)
Overall Behavioural $KU^{B}_{Planning(OffLoadLab)}$	Acceptable (a mean workload rating of 3.0, which is less than 3.3
Behavioural (associated with Considering Content) $KU^{B}_{C(OffLoadLab)}$	Acceptable (7.7 notepad entries last 2 lines of plan, is between 4 and 8)
Costs incurred by Laboratory Off-Load Planning Aids $rKC_{Planning(OffLoadLab)}$	
Structural $KC^{S}_{Planning(OffLoadLab)}$	Unacceptable (299 lines of code is too large an increase on 166, 40 interface objects is too large an increase on 22. But 1 handler calls 1 handler on other stacks, which is less than 2)
Behavioural $KC^{B}_{Planning(OffLoadLab)}$	Acceptable (40 mins. 6 secs. taken to produce a plan, is less than 42mins. 54secs.)
Behavioural (associated with Considering Content) $KC^{B}_{C(OffLoadLab)}$	Acceptable (1.5 secs. estimated run time, is less than 2.5 secs.)

Task Quality

The object_Scope of the off-load plan is acceptable. Indeed, it is better than desired - significantly fewer clerical errors are made with OPS1 than with OPS0.5. Better scoping of the plan accounts for approximately 50% of the improvement in plan content, and only increases computer costs by two interface objects and 25 lines of code. The time_Scope of the off-load plan, in contrast, is unacceptable in that, on average, only 6.4 lines of the off-load plan were available by the deadline (which is less than the 7 lines required). OPS1's considering content behaviours are risky, in that they fail to guard against the possibility of a plan that does not specify goals for the complete time period. If a goal for the planning task is expressed as a deadline, and an off-load plan is considered and produced one load at a time, and the time_Scope of the plan is extended as lines/loads are added, then whenever a student subject limits the rate at which they incur costs (works too slowly), the result is likely to be a poorly time_Scoped plan - specifically, a plan that fails to cover the full period. Alternative behaviours may make the production of an incomplete plan less likely. For example, suppose that the computer may consider the content of a plan for the complete time period, and make a plan for the complete period available immediately. Then, the student subject could further select, at will, any load from a draft off-load plan, and, if necessary, replace it with a better alternative. As a draft plan is modified in this way, the computer automatically re-considers the whole plan, revising any loads affected by the subject's modification. With such behaviours, whenever a student subject works too slowly, the result is likely to be a plan for the full time period with sub-optimal content in other respects (some loads may not have been fully considered by the deadline), rather than a poorly scoped plan.

The content of the off-load plan in terms of specified lift is lower than that achieved with OPS0.5, but still higher than desired (overloading craft is not effective). There appear to be two reasons: (i) student subjects assess content inappropriately - they sometimes fail to mark down overloaded craft. (Student subjects had yet to acquire the structures (representations of domain knowledge) for considering content); and (ii) having assessed alternative load options appropriately, student subjects add the wrong load option to the plan in error. (Once added to the plan, students could not 'undo' their decision and consider the options further).

User and Computer Behavioural Costs Associated with Considering Content
User behavioural costs associated with considering content are not significantly different from the desired level. Student subjects said that a particular load option assessment display (the one of all the scores of all the loads) helped them to consider content rapidly and accurately. It also helped them to learn how to consider content during the practice session. This display incurs relatively few computer costs (1 interface object and 5 lines of code), but evidently enables subjects to consider content systematically and economically (see section 3.3.1.3). Estimated computer behavioural costs associated with considering content are approximately comparable to those incurred by OPS0.5, and so are as desired.

Overall User and Computer Behavioural and Structural Costs
Overall user behavioural and structural costs are acceptable. (They are not significantly higher than those incurred with OPS0.5.) Subjects scored equally well on the multiple-choice tests set during training. Overall computer behavioural costs are acceptable. Student subjects took an equivalent amount of time to explore OPS1 as to explore OPS0.5 and subjective workload ratings are also similar. Overall computer structural costs have increased to an unacceptable level, however, and suggest a waste of computer resources. The computer's translation and comparison behaviours incur considerable costs (an additional 9 interface objects and 50 lines of code), but have little impact on task quality. The student subjects refer to verbal assessments and comparisons rarely, if at all, and so OPS1 appears to have some redundant facilities.

In summary, this assessment suggests that OPS1 has only partially achieved its required level of performance. OPS1's performance is as required with respect to scope, overall user structural and behavioural costs, and user behavioural costs associated with considering content. But its performance with respect to plan content and overall computer structural costs is not as required.

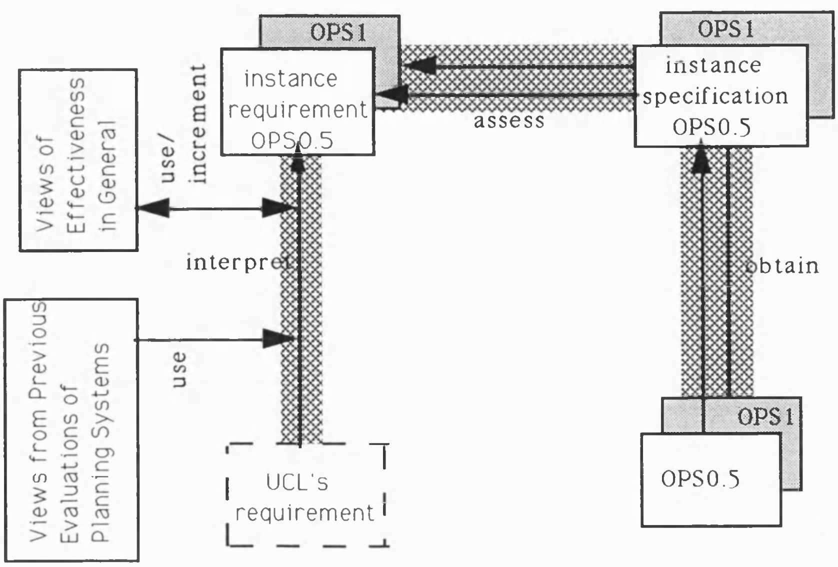
3.2. Explicit Craft Evaluation

This section briefly illustrates the conventional explicit Craft evaluation which, it is assumed, would have been conducted, had there been no informal Engineering alternative. The explicit Craft evaluation to be presented here is conducted within the general position of 'usability engineering' (Whiteside et al 1988; Gould, 1987; Gould & Lewis, 1987). The principal characteristics of late, summative evaluations conducted within such a position are taken to be the following: (i) a multi-disciplinary design team (Singleton, 1987); (ii) participation of end-users and other stake-holders (Norman & Draper, 1986; Mumford, 1979); (iii) explicitly stated and agreed usability goals expressed as concepts and indices (a 'usability specification')(Carroll and Rosson, 1985); (iv) context-sensitive interpretation and evolution of usability goals; and (v) awareness of relevant literature. During an explicit Craft evaluation, each member of the design team (including end-users) typically reviews their disciplines for relevant concepts and indeces in the context of the current prototype and its use. The design team then negotiates a mutually acceptable statement of usability goals, and assess the extent to which these goals have been satisfied. Explicit Craft evaluation of planning systems is exemplified by Mulvehill's evaluation of a satellite construction scheduling (1988) and by Ahmad et al.'s evaluation of network planning in the water industry (1988).

3.2.1. Application Representations

The application representations that support the equivalent explicit Craft evaluation of OPS1 are, first, the views of planning system effectiveness used in previous evaluations, and, second, views of effectiveness in general (see Figure 3.11).

Figure 3.11: Use of Other Views of Effectiveness to Support Late Evaluation: the case of OPS1



3.2.1.1. Other Views of Planning System Effectiveness

The effectiveness of previously evaluated planning systems has been viewed from a number of distinct perspectives. These perspectives reflect the loosely defined intellectual positions of the interested parties. The perspectives adopted below include 'human factors', 'software engineering', 'systems' and 'the end-user' perspectives. For simplicity, the review focuses on planning systems and military planning systems in particular.

Manufacturing Scheduling :
human scheduling performance, tardiness, plant utilisation (Sanderson, 1989);
intuitiveness, consistency, differential usefulness of displays, schedule feasibility, tardiness, plant utilisation (Pinedo et al, 1993)

Strategic Land Battle Planning :
thoroughness, foresight, contingency planning, imagination, boldness, technological exploitation, rigidity, detailedness (Reason, 1991)

Fleet Employment Scheduling:
fleet readiness and morale; level, necessity and equability of employment; and opportunity for training and maintenance (Brown et al, 1993)

Maritime Route Planning:
time on task (Tainsh, 1992)

Supporting Arms Target Allocation Planning:
weapon effectiveness, simplicity and flexibility of interaction (Slagle & Hamburger, 1985)

SSGW Planning:
good target information, surprise, concentration of fire, clearance (from section 2.2.2.2.)

3.2.1.2. Views of Effectiveness Generally

For Shackel, when viewed from a human factors perspective, evaluation problems involving human-computer interaction address the following aspects of usability:

(i) effectiveness
- at better than some required level of performance (e.g. in terms of speed and errors);
- by some required percentage of the specified target range of users;
- within some required proportion of the range of usage environments;

(ii) learnability
- within some specified time from installation and start of user training;
- based upon some specified amount of training and user support;
- within some specified re-learning time each time for intermittent users;

(iii) flexibility
- with flexibility allowing adaptation to some specified percentage variation in tasks and/or environments beyond those first specified;

(iv) attitude
- within acceptable levels of human costs in terms of tiredness, discomfort, frustration and personal effort;
- so that satisfaction causes continued and enhanced usage of the system.

(after Shackel, 1986)

Jordan et al. view usability somewhat differently (1991). For them, usability comprises guessability (the chances that a user will use a facility correctly at the first attempt), learnability (the time and effort required to reach a user's peak level of performance with a system) and experienced user performance (the level at which a user's performance with a system tends to flatten out after a period of time). In addition, Jordan et al. suggest, usability may also comprise discoverability (how much of the system potential the user exploits on reaching the asymptotic level) and re-usability (the time and effort required to re-achieve the asymptotic level of performance following a period of absence from the system).

Viewed from a software engineering perspective, software quality comprises a system's functionality, inter-operability, survivability, availability, extendability, software performance, maintainability, security, safety, reliability etc. (Boehm et al., 1978).

3.2.2. Use and Incrementation of Application Representations in Evaluation

3.2.2.1. A View of OPS1's Effectiveness

From some perspectives and to some extent, OPS1 may be regarded as similar to manufacturing scheduling systems, in that both concern scheduling. In the same way that manufacturing schedules may be evaluated with respect to the average tardiness of jobs and the utilisation of plant as a proportion of its maximum, off-load plans may be evaluated with respect to the average tardiness of men and equipment arriving ashore and the utilisation of landing craft and helicopters. OPS1 may also be regarded as similar to maritime route planning. In the same way that maritime planning may be evaluated with respect to the time spent planning a route for ships, off-load planning may be evaluated with respect to time spent planning a route for assault craft and helicopters. Combination of these concepts, produces the following view of the effectiveness of OPS1:

tardiness of arrival
utilisation of assets (assault craft and helicopters)
time on task

However, working within the Craft, it is difficult to specify further the relationship between OPS1, manufacturing scheduling systems and maritime planning systems, and the selectivity of the views of effectiveness devised for previous evaluations. That is, it is not clear what has been omitted from the previous views of manufacturing scheduling effectiveness and maritime route planning effectiveness, because of the specific purposes of the evaluations that the views supported. Consequently, there is little reason to believe that simply copying a few views will be sufficient for the evaluation of OPS1[5].

---

[5]Simply carrying forward a few views of effectiveness and re-using them in a subsequent evaluation is particularly problematic for end-user's views, since it threatens loss of validity and lacks precedent. If an end-user's view of effectiveness is simply re-used, then its validity is lost, because the view is no longer based on the knowledge of the relevant user. In the initial evaluation, such a view is based on user knowledge, but, in subsequent evaluations, such a view is just based on the view of another, interested individual. In this sense, carry forward of end-users' views

Another way of reasoning about the completeness or, rather, appropriate selectivity, of the view of effectiveness devised for the evaluation of OPS1 is required - conceiving effectiveness from multiple perspectives for the purposes. By increasing the number of perspectives on effectiveness, it is expected to increase the number of issues likely to be raised. By considering the purposes of the evaluation, it is expected to ensure that only relevant issues are addressed.

To develop a view of effectiveness appropriate for the evaluation of OPS1, then, it is necessary to use many specific and general views of effectiveness as starting points for reasoning about the evaluation at hand, and to regard such views as useful if interpreted appropriately for the purposes. There is no expectation that any one such view will be complete or coherent for the purposes of evaluating OPS1. Rather, carry forward seeks views that are explicit and complement each other well, so that, when interpreted in the context of the evaluation of OPS1, a sufficiently wide range of issues may be raised, and the relevant issues selected[6]. Reasoning

---

without further reasoning for the purposes threatens a return to the discredited practice of 'presumption guided by common sense'. Also, the spread of military planning systems suggests that views of quality have been acquired from end-users in the manner of section 2.2.2. many times before and that effective enough systems have resulted. However, there is little precedent for 'ask one user, then simply re-use the view'. Carrying forward some mature Craft views of effectiveness in anything other than an opportunistic, purpose sensitive manner, remains to be demonstrated and argued for.

[6]It may be argued that the mature Craft evaluation would be more representative of current practice if, following the approach of user-centred design, the view of OPS1's effectiveness took the end-user as its starting point rather than previously used views reported in the literature (Norman & Draper, 1986 p.2). In section 2.2.2.2, the view of effectiveness obtained through end-users was reported as 'a simple, rapid and prioritous landing' and the end-user's view does indeed influence the mature Craft formulation of the instance requirement (see later this section). Norman & Draper's approach was not adopted here since it highlights 'going back to

about other views of effectiveness also seeks to resolve, for the purposes, discrepancies between incommensurate perspectives[7].

For the purposes of illustration, the product of such reasoning is assumed to result in an instance requirement that equates closely to the requirement developed by the informal Engineering evaluation (see Table 3.6). Explicit Craft and informal Engineering do not always result in identical design products. However, there is no reason why such similarity should not hold in this case.

Thus, given the focus of the evaluation and the metrics utilised (see section 3.1.), the effectiveness of OPS1 may be expressed in terms of: 'task completion' (after Tainsh's 'time on task' and Shackel's 'effectiveness'); 'simplicity/speed/prioritousness of landing' (from the end-user and Shackel's 'effectiveness'); 'explorability' (after Jordan et al.'s 'guessability' and 'learnability' and Shackel's 'learnability'); 'workload' (from general

---

the beginning again' for every system development project rather than carry forward.

[7]For example, both the terms 'usability' and 're-usability' have different connotations when employed within user-centred and device-centred perspectives (see Boehm et al. 1978; Shackel 1986; Jordan et al. 1991). Applied from a user-centred perspective, the concept of 'usability', where it may comprise, for example, learnability, flexibility and attitude, has a broader meaning than when applied from a device-centred perspective, where it may connote only one of many 'non-functional requirements'. Applied from a user-centred perspective, the concept of 're-usability' may refer to the ease with which a user re-learns or remembers how to interact with a system after a period of non-use. Applied from a device-centred perspective, 're-usability' may refer to the ease with which a developer may 'cut and paste' code written for one system into another system. Incommensurate perspectives may adversely affect evaluation in a variety of ways. For example, the outputs of separate HF and SE evaluations may be conceptually incompatible, or communicated poorly, and may lead to misunderstanding between evaluators, delay, or duplication. Also, trading off the different implications of HF and SE evaluations for any re-design of the system may be difficult or impossible. A view of OPS1's effectiveness for the purposes seeks to resolve such incommensuracy in the instance.

psychology and Shackel's 'attitude'); 'reasoning strategy' (after analysis of OPS1 and Shackel's 'attitude'); 'clerical errors' (after analysis of OPS1 and Shackel's 'effectiveness'); 'efficiency and concise-ness of coding; and maintainability' (after Boehm et al.'s maintainability); and 'response time of load generation and assessment algorithm' (after analysis of OPS1). To reflect the perspectives adopted, these concepts are categorised in terms of Human Factors and Software Engineering issues.

Human Factors Issues

The utility of OPS1 is mixed. On average, subjects spent an acceptable amount of time on the task (40 mins. 6 secs.); planned an adequately simple, speedy and prioritous landing (a rate of lift 267 men/hr.); and made relatively few clerical errors (0.4 errors). However, one subject failed to complete the task within the deadline (hence, on average, only 91.5% of the task was completed by the deadline). The usability of OPS1 is acceptable. Subjects explored the system easily (only 32 mins. 31 secs. were required to explore the system to their satisfaction) and appeared to learn enough during exploration (subjects scored 15.9 correct answers on the post-exploration test). Further, subjects' workload during the task was acceptable (the mean workload rating was 3.0), employed an adequately efficient reasoning strategy (7.7 notepad entries for the last two lines of the plan) and made relatively few clerical errors (0.4).

Software Engineering Issues

The quality of OPS1 as software is mixed. OPS1 appears to be adequately maintainable (only 1 handler calls a handler on another stack) and to have improved the response time of the load generation and assessment algorithms (2.5 secs. reduced to 1.5 secs). However, the code appears to be inefficient and not concise - it now comprises 299 lines of code and 40 interface objects (instead of 167 lines of code and 22 interface objects).

OPS1, then, has only partially achieved its required level of performance.

This section concludes the illustration of a conventional, explicit Craft evaluation as a point of comparison for the informal Engineering evaluation. The following section assesses the relative effectiveness of the informal Engineering and explicit Craft evaluations with respect to the case reports and considers how well carry forward in a manner characteristic of informal HCI Engineering has been illustrated.

Table 3.6. Indices of Human Factors and Software Engineering Aspects of the Effectiveness of Laboratory Off-Load Planning Systems

Concept	Index
Human Factors Aspects	
Task Completion	mean percentage of plan completed by the deadline (%)
	mean time to produce a plan
Simplicity/Speed/Prioritousness	mean planned rate of lift (men/hr.)
Explorability	mean duration of exploration mean no. of correct answers on test
Workload	mean workload rating
Reasoning Style	mean no. of notepad entries for the last 2 lines of plan
Clerical Errors	mean no. errors in Columns 2,3,4, & 10 of plan
Software Engineering Aspects	
Efficiency & Concise-ness of Coding	lines of code (excluding initialisation, and data collection) no. interface objects
Maintainability	no. handlers that call handlers on other stacks
Response Time of Load Generation and Assessment Algorithms	estimated run time (standard interaction)

3.2.2.2. The Conformance Between Actual and Required Effectiveness of OPS1

The informal Craft evaluation of OPS1 is illustrated using the same data and conducted with the same purpose as the informal Engineering evaluation. Consequently, the explicit Craft statements of required and actual effectiveness and their conformance are essentially identical to the informal Engineering statements of required and actual performance and their conformance, but expressed within different conceptual frameworks[8]. To simplify the illustration, only the explicit Craft assessment of conformance in considered in full (see Table 3.7).

3.3. Assessment of Case Report

To summarise the outcome of different manners of carry forward in evaluation, the informal Engineering and explicit Craft alternatives are shown in Figure 3.12.

3.3.1. Scope and Content of Evaluations

In the informal Engineering evaluation, as a result of adopting a domain-oriented, systems perspective, issues conventionally conceived as HF and SE issues have been conceived as unified HCI issues. For example, the evaluation considered HF issues, such as OPS1's usability, learnability and utility. In conventional terms, it suggested that relevant information (all the scores of all the options) was easier to access with OPS1, and so learning was more systematic and effective. This outcome had consequences for the utility of OPS1 because, when subjects had to plan under time pressure, they were more able to plan efficiently and calculate the best load correctly. In the informal Engineering evaluation, this issue was expressed as an improvement in the quality of plan content due to a reduction in user behavioural costs associated with considering content, and for a minimal increase in computer structural costs, that is, in terms of an explicit trade-off between usability, learnability and utility and the implications for the computer. As another example, in the explicit Craft evaluation, it was suggested that requiring a plan to be produced one line at a time, and to be available by a deadline, risks the production of an unavailable (late) plan - an incomplete task. A preferable alternative may

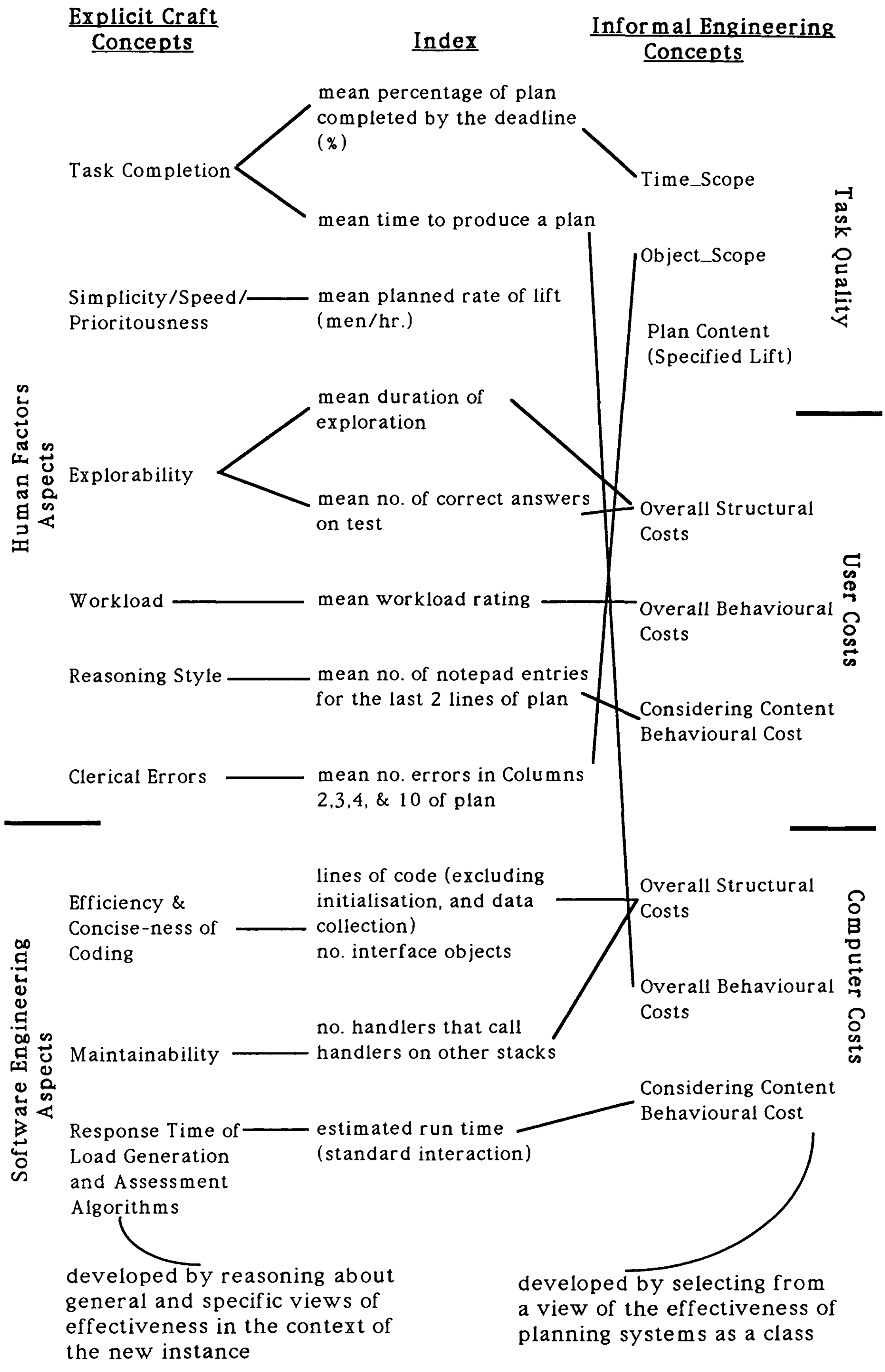
---

[8] Perception of identity is dependent upon agreement that the author's mapping between craft and engineering concepts is an acceptable one.

Table 3.7. An Assessment of OPS1's Effectiveness from a Human Factors and a Software Engineering Perspective

Concept	Index
Human Factors Aspects	
Task Completion	Unacceptable (91.5% available by the deadline is less than 100%)
	Acceptable (40 mins. 6 secs. taken to produce a plan, is less than 42mins. 54secs.)
Simplicity/Speed/Prioritousness	Acceptable (a planned rate of lift 267 men/hr. is between 255 and 275 men/hr.)
Explorability	Acceptable (32 mins. 31 secs. exploring, is less than 35 mins. 39 secs.)
	Acceptable (15.9 correct answers on test, is not less than 15.9)
Workload	Acceptable (a mean workload rating of 3.0, which is less than 3.3
Reasoning Style	Acceptable (7.7 notepad entries last 2 lines of plan, is between 4 and 8)
Clerical Errors	Acceptable (0.4 errors in Columns 2,3,4, & 10 is less than 1.8)
Software Engineering Aspects	
Efficiency & Concise-ness of Coding	Unacceptable (299 lines of code is too large an increase on 166, 40 interface objects is too large an increase on 22.
Maintainability	1 handler calls 1 handler on other stacks, which is less than 2)
Response Time of Load Generation and Assessment Algorithms	Acceptable (1.5 secs. estimated run time, is less than 2.5 secs.)

Figure 3.12: A Summary of Alternative Manners of Carry Forward in Evaluation: the Case of Late Summative Evaluation of OPS1



be to require the production of a complete draft plan early on, and the gradual refinement of the complete draft. In the informal Engineering evaluation, such issues were expressed as alternative implementations of 'making available and considering content' behaviours and their relative advantages for achieving certain kinds of availability goals.

The informal Engineering evaluation also considered SE issues, such as OLP1's modularity and functionality. In the explicit Craft evaluation, it suggested that the isolation and independence of different parts of the program was greatly increased by the separation of the 'assessment' and 'detailed option display' scripts. It also questioned the functionality of the translation and comparison scripts, because these facilities were rarely used. In the informal Engineering evaluation, these issues were expressed as a reduction in computer behavioural costs incurred when considering content and an increase in overall computer structural costs for little impact on plan content.

Both evaluations expressed the effectiveness of OPS1 relatively concisely, and had expressed a similar assessment of OPS1 - that it satisfied the requirement only in part.

3.3.2. Reasoning About Completeness/Selectivity

The informal Engineering evaluation differed from the explicit Craft evaluation in terms of how the evaluation problem was conceived and so how the completeness and/or appropriate selectivity of the evaluation was reasoned about. The informal Engineering evaluation conceived the Problem instance addressed as an instance of a class of evaluation Problem (see section 1.1.4). Consequently, the completeness of the OPS1 evaluation could be reasoned about with respect to the expression of the evaluation of military planning systems generally. For example, since the class indicated that an evaluation should assess the aspects of effectiveness 'a, b and c', then, in any instance, a 'complete' evaluation would be expected to assess instantiations of 'a, b and c'. In this case, the evaluation addressed two aspects of Task quality (scope and content), but did not address another (view). The class approach fails if the conception of the class of evaluation problems is inadequate - if the class is conceived incompletely, then the instance will be also. In this case, however, the expression of the general military planning system evaluation problem.

In contrast, the explicit Craft evaluation addressed OPS1 as a problem instance related to other instances in poorly specified ways (see section 1.1.4.) - off-load planning was just 'like' manufacturing scheduling and maritime route planning. Also, the selectivity of views of effectiveness used in previous evaluations is poorly specified. For example, if a previous evaluation assessed aspects of effectiveness 'x, y and z' and was not obviously incomplete, so the forth-coming evaluation of OPS1 (which is presumed to be like the previous case in some unspecified ways) should also assess aspects of effectiveness 'x, y and z'. However, since the previous and forth-coming evaluations are not fully conceptualised, it is difficult to assess how complete the forth-coming evaluation will be. To develop an acceptable reason to believe that the view of effectiveness used would be appropriately selective, the explicit Craft evaluation applied multiple perspectives to the problem, in order to increase the number of issues likely to be raised.

It is important to note that reasoning about completeness with respect to classes of Problem is an addition to reasoning within multiple perspectives (see section 1.1.4.1). That is, in principle, a class of Problem may be conceived from a number of different perspectives and a number of expressions of the Problem class may be developed. In this illustration, the evaluation Problem addressed was formulated from a single, integrated domain-oriented, systems perspective because the class approach fails if the class expression is, itself, incomplete and a domain-oriented, systems perspective was considered to encourage complete Problem expressions by integrating Human Factors, Software Engineering and end-user concerns. However, had resources permitted, this illustration could have reasoned about the completeness and/or selectivity using both the class approach and the multiple perspective approach.

3.3.3. Adequacy of Illustration of Informal Engineering

Consideration of how well this report illustrates carry forward in informal HCI Engineering, suggests that informal Engineering was partially realised in that an application representation (the Problem hierarchy) was incremented as a result of its use (see section 1.1.3). Also, knowledge about classes of design Problem and instances of classes of design Problem was applied to support the development of an instance requirement and an instance specification. an informal class of Problem was formulated, and an instance of this class of Problem was addressed. Further, the additional means of reasoning about the completeness of design representations provided by informal Engineering was used (see section 1.1.4.). However, general (class) requirements and general (class) specifications were presumed rather than made explicit and the artefact was not conceived as an instance of a class of artefact - OPS1 is a bespoke implementation, which is presumably related to other implementations, but in unstated ways. Design work only comprised a transformation from artefact to instance specification, and from client requirement to instance requirement, followed by an assessment of the instance specification with respect to the instance requirement, and so opportunities for carry forward in addition to those available within explicit Craft were not, in actual fact, fully created or exploited.

Carry forward was partially realised in that an expression of the general planning system evaluation Problem was developed from the pre-conception and the HCI(e) conception, and repeatedly instantiated, first, for the evaluation of OPS1 reported in detail, and second, for the putative evaluation of some un-named SSGW planning system alluded to in Section 2.2.2). However, re-use was not fully carried through (see section 1.1.3). The SSGW planning system evaluation Problem is not fully instantiated, only that part of the expression that concerns SSGW plan quality. Further, an SSGW worksystem is not, in actual fact, evaluated, so the effectiveness in practice of the informal Engineering approach remains, with respect to SSGW planning, a matter for speculation. Further, since the knowledge carried forward supported the late evaluation of a prototype, that is, a transformation from artefact to instance specification, from client requirement to instance requirement, followed by an assessment of the specification with respect to the requirement, opportunities for carry forward in addition to those available within explicit Craft were not, in actual fact, created or exploited.

3.3.4. Relative Effectiveness of Informal Engineering Evaluation in Practice

Overall, judgements about the relative effectiveness in practice of the informal Engineering and explicit Craft late evaluations rest upon: (i) the perceived value of the savings of time and effort achieved by reasoning about completeness with respect to classes of Problem, rather than with respect to other problem instances; and (ii) the perceived value of the additional investment required to develop the pre-conception, Problem hierarchy and Problem expressions, rather than document case histories and form general views of effectiveness (see section 1.2.1). The explicit Craft means of encouraging completeness is judged here to be not especially difficult. In essence, it required reviewing the literature, talking to users, and considering the artefact in the instance. It was as difficult, in fact, as specialising the general Problem expression for the instance, and selecting an appropriate focus for the evaluation. In contrast, the development of the pre-conception, Problem hierarchy and Problem expressions required considerable effort. Overall, the cost of developing the pre-conception etc. seems not to be justified by the ease of reasoning about completeness, at least in this instance, and relative to the explicit Craft alternative.

Such a judgement, of course, rests upon a number of assumptions. First, it is assumed that the informal Engineering and explicit Craft evaluations are of approximately equal value in terms of their support for the development of subsequent user requirements. That is, they are equally insightful about planning effectiveness and the the state of OPS1. After all, the informal Engineering view of effectiveness derives, ultimately, from the pre-conception, which was developed following methods similar to those used to develop the explicit Craft view of effectiveness. Given similar methods, equal 'insightfulness' is, perhaps, to be expected. However, it may be argued that, in this case, such an assumption may be unfounded. Rather, since in explicit Craft, an appropriate selective view of effectiveness emerges as a result of discussion with interested parties and of review of other examples, explicit Craft may indeed achieve equivalent views of effectiveness, but only later, rather than immediately. For example, perhaps some aspects of effectiveness would only come to light during discussions with users in the context of a prototype. If prototyping was required to develop a view of effectiveness, the explicit craft approach may be less attractive. In this case, an equivalent to the pre-conception's concept of 'plan view' or representation was not raised by end-users, or indeed other specific or general views of effectiveness. In the case of off-load plans, end-users tended to focus upon what the pre-conception referred to as plan content, rather than plan scope or plan view (see

Section 2.2.2.2). From an end-users' view, off-load plans produced by hand take a traditional form and just 'are the way they are and about what they are about'. Previous evaluators also appear not to have addressed this issue. Although it is inconceivable that a planning system could be repeatedly evaluated without the issue of 'what the plan produced looks like' coming to the fore, such a concept, in this case, did not do so immediately. Second, conducting explicit Craft evaluations on one's own for purposes of research may be less difficult than conducting them in a team and for actual development purposes. Specifically, resolving incommensurate perspectives in the instance may be more difficult when a multi-disciplinary team is involved. For example, a team of evaluators addressing actual evaluation problems may need to spend considerable time acquiring an understanding of each other's perspectives, concerns and skills. They may also need to meet and discuss issues regularly, and become part of a single design team (Gould 1987; Hutt et al. 1987). Standard report formats may be useful; or specific, objectively measurable design goals may be agreed (Whiteside et al. 1988). Should there be serious disagreement, then an explicit conception of the evaluation being conducted may have to be negotiated. There are costs, then, to coping with incommensurate perspectives.

Thus, this initial attempt to realise carry forward in a manner characteristic of informal HCI engineering is only partially successful. Subsequent attempts, then, must illustrate carry forward more fully, ensure that a class requirements and specifications be constructed, and develop application representations that provide greater savings in development time and effort when designing instances.

Chapter 4.
Second Attempt: Carry Forward in Specification

Summary
This chapter reports an attempt to realise carry forward in *specification* in a manner characteristic of informal HCI Engineering. The attempt is compared and contrasted with the explicit Craft alternative and we then consider the adequacy with which the desired manner of carry forward is illustrated. This Chapter also seeks to make good the deficiencies of the initial attempt at carry forward reported in chapter 3.

In the informal Engineering evaluation, the application representations involved are the Problem hierarchy (see Chapter 3), the pre-conception, (see Chapter 2), Dowell & Long's general design Problem (see Chapter 3), a general (class level) Problem element, which concerns initiating planning interactions, and a general (class level) Graphical User Interface object, which specifies a menu structure for planning systems. First, a general (super-ordinate class) menu structure and its associated general (super-ordinate class level) Problem element are developed, with reference to existing Graphical User Interfaces (GUIs) (the artefacts), analyses of GUIs, and Dowell & Long's expression of the General Design Problem. Second, the general (super-ordinate level) menu structure and general (super-ordinate level) Problem element are specialised for military planning, with the support of the pre-conception and a brief, domain-oriented analysis of planning worksystem behaviour. Relevant guidelines for the design of menu structures are applied during the design of general menu structures of both the super-ordinate class level, and the class level. The design work to which the application representations are applied concerns, first, the development of OPS1, UCL's Off-Load Planning System, and second, SATCON1, UCL's Satellite Construction System. Like OPS1, SATCON 1 is a reconstructed, simplified system, developed for the purposes of the research. For each system, a client requirement is presumed and an instance requirement is developed in the form of a statement of required performance. Selected GUI objects, such as plan views, and view filtering and formatting dialogues, are specified. For each system, a partial prototype is then implemented. In each case, specification is supported by reviews of relevant demonstrator systems. The use of the application representations

in specification begins with the interpretation of UCL's problems (concerning OPS1 and SATCON1) as two instances of the general Problem of military planning systems.) This categorisation of UCL's problem increments the Problem hierarchy through the addition of a new sub-sub class (concerning military satellite construction) and prompts the development of a general requirement for military planning systems with GUIs. In response to this requirement, a general specification is then developed by specialising and integrating the available and relevant general GUI objects. At the time the work was conducted, the only such objects available were: (i) the general (class level) menu structure for planning tasks developed earlier in this Chapter; and (ii) a general (class level) menu structure for personal file control tasks[1]. The general specification is then instantiated, first, for OPS1, and second for SATCON1. with the support of domain and worksystem analyses of the respective instances. This instantiation is supported by the instantiation of the domain pre-conception and brief worksystem analysis for off-load planning at UCL and military satellite construction at UCL.

In the equivalent, explicit Craft specification, an identical design context (partial specifications of OPS1 and SATCON1) is assumed. The application representations are specifications of menu structures of other planning systems, and guidelines formulated from various perspectives and at various levels of generality. The use of the application representations begins with the perception of a similarity between OPS1 and BATTLE, a weapon allocation planning system developed by Slagle and Hamburger (1985), and the incorporation of BATTLE's menu structure in the instance specification for OPS1. However, the relationship between OPS1 and BATTLE, and the selectivity of BATTLE's menu structure are poorly specified. As such, there is little reason to believe that simply copying any one menu structure will be sufficient for OPS1. Consequently, elements of many previous menu structures are selectively re-used, and design guidelines are interpreted for the instance. A similar process of selection and interpretation is presumed for the specification of SATCON1's menu structure.

---

[1] Since the work was conducted, additional relevant general (class level) objects for planning tasks have come to light, specifically for plan views and plan buttons (Van Putten et al., 1993) and so the general specification may be expected to become increasingly complete with time.

Comparison between the products of informal Engineering and the explicit Craft specification suggests that, in this case, the former is likely to result in interactions which are more consistent, which may benefit users who wish to interact with more than one planning system by enabling the transfer of knowledge and skills relating to one planning system to other planning systems. In other respects, the assessment of the specifications were inconclusive, since it was difficult to judge whether a perceived need to re-design the prototypes was attributable to poor specification (and, by implication, poor general specification and GUI objects), the selectiveness with which the Problems were addressed, or the inaccurate presumption of requirements. However, assessment failed to establish that the instantiations of the general specification were *ineffective* relative to the crafted menu structures of equivalent demonstrator systems. The informal Engineering specification was different to the mature Craft specification in terms of how the specification was conceived and how the completeness of the specification was reasoned about.

Consideration of how well this report illustrates carry forward in informal HCI Engineering suggests that the deficiencies of the initial attempt have been remedied. Re-use was fully carried through - the pre-conception of the domain of C^2 supported, first, the domain analysis of off-load plans, and second, an analysis of satellite construction schedules. Also, design work involved the abstraction of a general requirement, the development of a general specification and the instantiation of this general specification. The development of a class specification was also supported by knowledge of classes of design Problem - the general Problem elements and the general GUI objects. In addition, and as in the illustration of carry forward in evaluation, the additional means of reasoning about the completeness of design representations provided by informal Engineering was used, an informal class of Problem was addressed, and a knowledge representation - the Problem hierarchy - was incremented. Thus, overall, additional opportunities to apply knowledge were, in this case, created and exploited.

Overall, judgements about the relative effectiveness in practice of the informal Engineering and explicit Craft specifications rest upon: (i) the perceived value of the savings of time and effort achieved by reasoning about completeness with respect to general specifications, rather than selectively re-using other instance specifications and interpreting guidelines in the instance; (ii) the perceived value of the additional investment required to develop the pre-conception, Problem hierarchy Problem expressions, general Problem elements and general GUI objects, rather than documenting design work and developing design guidelines; (iii) the perceived value of additional opportunities to apply knowledge to design work; (iv) the perceived value of general specifications, that is, the design of an infinite number of instances, rather than the instance at hand; and (v) the perceived value of user interfaces that are consistent across planning tasks. In this case, and in the author's opinion, if *complete and coherent* general (class) specifications may be developed in response to *actual* requirements, then the cost of developing the pre-conception etc. may be justified by the additional opportunities, ease of reasoning about completeness and consistency. That is, if it may be scaled-up, and transferred to real world Problems, carry forward in this manner may constitute a worth-while advance.

Further work, it is suggested, should seek to 'scale-up' the manner of carry forward in specification illustrated here and to apply it to actual Problems.

4.1. Informal Engineering Specification

4.1.1. Development of Application Representations

In this attempt to realise carry forward in specification in a manner characteristic of informal HCI Engineering, two additional application representations are developed - a general (class) menu structure for military planning tasks - and its associated general (class) requirement element - the element of initiating planning interactions. First, a general (super-ordinate) menu structure and its associated general (super-ordinate) requirement element are developed, supported by an appreciation of user interface styles, analyses of user interface style objects, and Dowell & Long's expression of the General Design Problem. Second, the general (super-ordinate) menu structure and associated general (super-ordinate) requirement element are specialised for planning tasks, with the support of the analysis of the planning domain within the pre-conception of C^2. Relevant guidelines are applied to support the specification of both the super-ordinate and class level menu structures (see Figure 4.1).

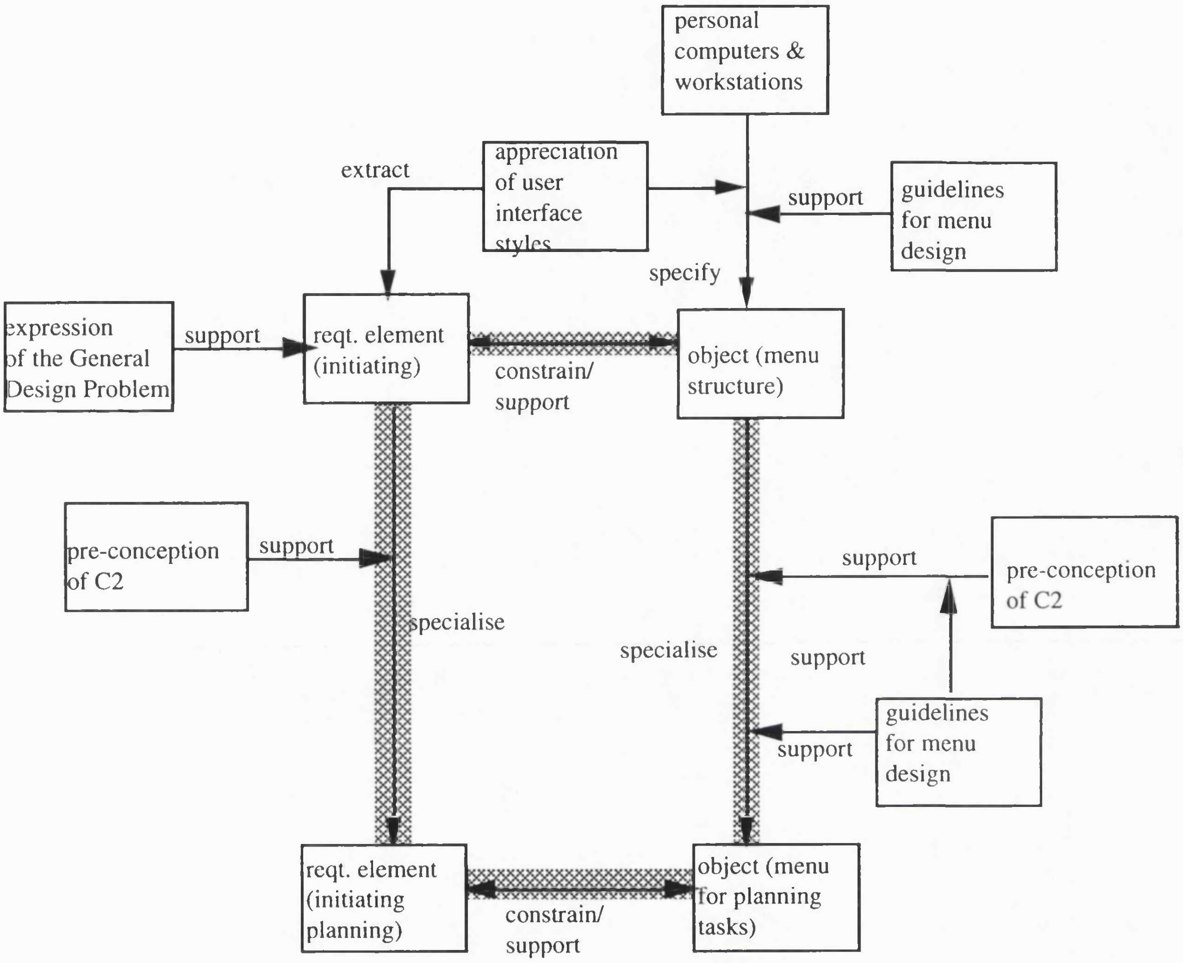
4.1.1.1. Rationale for Selection of Application Representations

The decision to develop a general menu structure for planning tasks as an application representation was based upon an appreciation of graphical user interface styles as implicated in a manner of carry forward that: (i) exhibits at least some of the characteristics of informal HCI Engineering; (ii) is effective enough in practice; but (iii) is deficient in terms of the support offered to design work. It was also influenced by design guidelines which were taken to suggest that, despite its limited scope and content, the pre-conception was nevertheless likely to support the specification of many aspects of a menu structure.

An Appreciation of User Interaction Styles

User interface styles, are increasingly popular. Initially, styles were designed by the vendors of computer platforms for their operating systems, for example, the Macintosh 'Finder' (Apple, 1985), Microsoft's 'MS Windows' (Microsoft, 1992) and Sun's 'Open Look' (Kannegaard et al., 1988). In recent years, end-user organisations and third party software

Figure 4.1. Development of a General Graphical User Interface Object - a Menu Structure for Planning Tasks



houses have designed interface styles to meet the requirements of general purpose computer users (for example, the Open Software Foundation's 'Motif' (OSF, 1991) and particular types of application, for example, military intelligence gathering systems (Braim & Hepworth, 1992) and telecommunications systems (Mahoney & Gower, 1991).

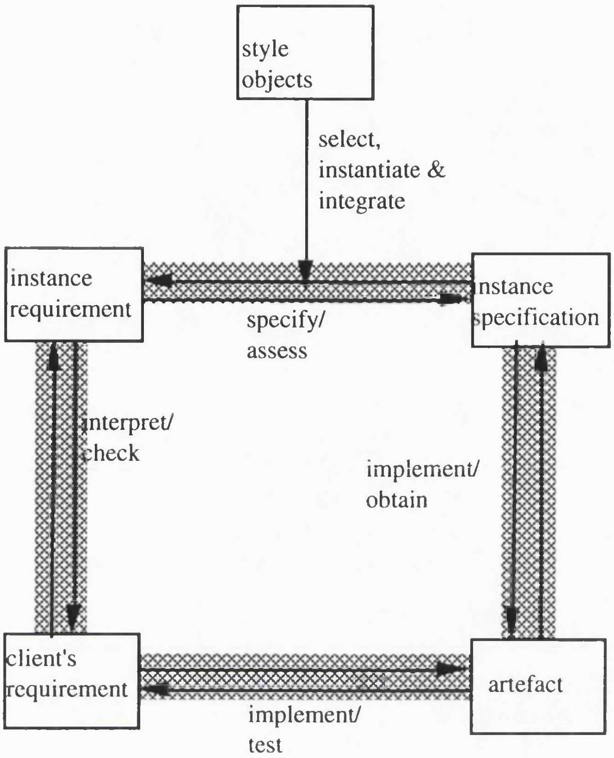
In this thesis, an interface style is taken to comprise a structured collection of general graphical user interface objects, which may be selected, specialised and integrated to develop an instance specification (see Figure 4.2). A general user interface object, here, is an element of a general specification, that is, a specification of a separable subset of the contributions to human-computer interactions in an instance. Separable, here, implies that one object may be re-specified without necessitating the re-specification of other objects in order to retain the consistency and coherence of the instance specification - the instance specification is modular. A GUI has been specified elsewhere as supporting interaction which the user experiences as directly and immediately engaging with a virtual world (Hutchins et al., 1985). For this thesis, a general *graphical-direct manipulation* user interface object is an object which supports computer contributions to interaction which are visual, rather than aural or tactile, for example, and user contributions to interaction which are perceived by the user as manipulative, rather than verbal or textual[2]. Thus, a GUI may be said to support computer 'displaying' behaviour and user 'manipulating' behaviour. Specifications of separable sub-sets of interaction, then, must specify computer displays and user manipulations (see Section 4.1.1.2).

To distinguish the term 'user interface style' from related terms, a style guide is taken to be a document in which object specifications are presented, together with guidelines for developing interfaces that comply with the style. An interface 'toolkit', 'environment' or 'User Interface Management System' (UIMS) (for example, Mahoney & Gower, 1991) is a programming environment in which interfaces which comply with the style may be implemented.

Currently, the majority of objects that comprise a user interface style - windows, menus, buttons, message bars, views, dialogue boxes, icons cursors etc. - tend to address the super-ordinate level of generality, that is, they specify the computer contributions to interactions for work tasks generally. Consequently, the majority of objects concern the 'look and feel' of interaction, that is, the fundamentals of mouse and keyboard

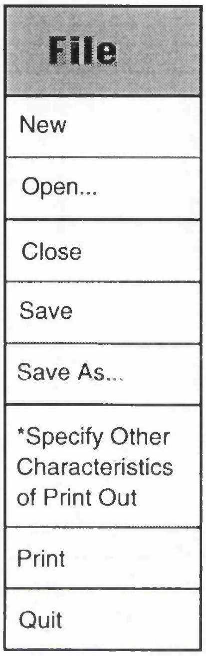
[2]Use of the term 'object' rather than 'specification element' is intended to emphasise the fact that the styles and objects of concern to this thesis are graphical. Styles currently tend to specify graphical-direct manipulation interactions. The potential range of interactions that may be specified at a general level, however, is far larger, and includes speech, 3-D imaging, gesturing, touch screens, to name but a few (Jacob et al., 1993).

Figure 4.2: User Interface Styles as Collections of Objects Supporting Design in the Instance



output and the physical appearance of the visual display (hence reference to a collection of objects as a 'style'). However, other objects that comprise a style address design at a more specific (class or sub-class) level, but specify more aspects of interaction. These objects reflect a different trade-off between generality and scope. For example, Rosenberg and Moran attempted to design a general (sub-class) menu for computer file control tasks (1982). (The modern equivalent of this menu, inferred from the personal computer workstations analysed in shown in Figure 4.3). (McCracken & Akcsyn, 1984, and Van Putten et al, 1993, have also attempted to design general objects, but at lower (class, sub-class or sub-sub-class) levels of generality.

Figure 4.3.: A General Menu for Computer File Control Tasks (inferred from Macintosh and MS Windows and after Rosenberg & Moran, 1982)



Key: * = place-holder, rather than actual manu label

Interface styles are thought to encourage the efficient design of effective systems in a number of ways:

(i) specifications and code are re-usable, and so design effort is saved. When the style is associated with a UIMS, implementation effort may be reduced by a factor of four or five and the amount of source code may also be reduced (Schmucker, 1987). For example, the Macintosh 'User Interface Toolbox' offers a set of routines that every Macintosh application may call as required. Each artefact, then, need not implement all aspects of its interaction from scratch;

(ii) the style itself is well designed, so minimising user error and difficulty (Bewley et al., 1983; Smith et al., 1982). Also, industrial designers may participate in style development and ensure that a style is attractive and projects an appropriate corporate identity (Gale & Brennan, 1993; Ohlfs, 1991);

(iii) styles enable consistent user interfaces. Consistent user interfaces are considered to be more familiar to a user and so easier to learn, and to increase user confidence (Kellogg, 1987; Barnard et al., 1981). It may also reduce the chances of confusion that may result from divergent design. Consistency is a widely applicable, and fundamental rule of thumb for interface design (Denley et al., 1992);

(iv) interface styles are easily transferred to, and acquired by, third parties. It is relatively easy to learn how to apply a style appropriately, particularly when the style is offered in conjunction with a toolkit, guidelines and an interactive tutorial (Alben et al., 1994). In addition, designers may learn about a style by analysing existing applications that are written within the style - a learning strategy that feels natural to many designers (Grief, 1985).

Such is the spread of interface styles, that many have come to be *de facto* standards for interface design (Buxton, 1993). Compatibility with an interface style is also perceived by many end-users to 'guarantee' a certain, acceptable level of usability and learnability. Interface styles, then, appear to satisfy a designer's need for support, and an end-user's need for 'guarantee' of effectiveness, at least to some extent, and better than other less widely adopted forms of 'design support'.

This appreciation of user interface styles, then, suggests that they be regarded as application representations that implicate the design practices of general informal specification and informal specialisation. As such, styles appear to exhibit at least some of the characteristics of informal Engineering (see Table 1.1). Since their spread may be interpreted as suggesting their effectiveness in practice, user interface styles may provide a useful starting point for attempts to illustrate this manner of carry forward.

However, the current tendency for interface styles to comprise mostly objects designed at the super-ordinate level for work tasks generally, may result in failure to support designers adequately. Some complain that the sub-set of aspects of interaction specified is too small and superficial.

"The style guidelines associated with most ... commercially available interfaces do little more than specify what classes of users' choice

operations should be supported by which of the supplied selection widgets. Typically, the domain areas of the supplied interfaces, where users actually do their work, are blank spaces with little more in the style guidelines to show how these should be filled."
Hakiel, 1993, p. 1

Further, a style begins to support design at a somewhat late stage, specifically, towards the end of detailed design and so fail to inform many of the more important design decisions that occur during earlier elicitation and analysis, and conceptual design (Lim et al., 1993).

There is a need to enhance interface styles, then, by designing objects at lower levels of generality (class or sub-class level).

Menu Structures and Domain Analysis

The second consideration which influenced the decision to develop a generic menu structure for planning tasks is the constraints and limitations imposed by supporting the design with preliminary knowledge. The nature of the pre-conception constrains the type of design support it may offer, and so constrains the type and level of generic object that is likely to be specifiable with acceptable effectiveness. A menu structure, here, is a set of menus and menu options which categorise, label and display alternative interactive behaviours and/or the work goals that such behaviours may achieve. Menu structures comprise the computer contribution to *initiation* interactions. When one party is in control of the interaction as a whole, typically the user, an initiation interaction is one that results in the on-set of another interaction (Paap & Roske-Hofstrand, 1988). Menu use, then, is taken to be an interaction aimed at determining subsequent interaction. Since menus 'categorise, label and display ..... *work goals*' (line 8, this paragraph), and the pre-conception comprises a narrow, high level characterisation of *work goals*, the pre-conception may be expected to support the design of menus.

More specifically, menu names (*labels*) that express work goals are likely to be effective because: (i) during menu use, the user is thought to formulate an intention in terms of a work goal and then matches this goal with a goal displayed as a menu-option label; and (ii) menus that display the goals that may be achieved through interaction with the computer are thought to help the user to develop a mental model of the device (Snyder et al., 1985). However, since menus also 'categorise, label and display ..... *interactive behaviours*', the pre-conception alone is unlikely support the design of menu labels completely. Users who formulate their intentions in terms of subsequent behaviour, or who seek to develop a mental model based on the system's behaviour, may prefer menus whose labels directly express these behaviours. Thus, Paap and Roske-Hofstrand's guideline for menu labelling reads:

"The name or phrase used to designate each [menu] option should be precise. The name should permit the user to infer precisely those actions or objects that are controlled by the selection of the options without missing anything that should be included or including anything that is extraneous."
p. 216

Similarly, menu categorisations that reflect work goals are also likely to be effective, since an effective categorisation is thought to match the user's 'conceptual organisation' and a user's conceptual organisation is likely to reflect work goals and 'their understanding of the task' (Barnard & Grudin, 1988, p249). However, since user's 'understanding of the task' is likely to include both work goals and interactive behaviours, the pre-conception alone is again likely to support menu design only in part. Given the intentionally preliminary nature of the pre-conception, partial support may be the best that may be obtained.

4.1.1.2. Development of a GUI Menu Structure for Work in General and its Associated General Problem Element

In this Section, a general (super-ordinate) menu structure and its associated general (super-ordinate) Problem element are developed. First, the Macintosh and MS Windows GUI styles were analysed and a general GUI object - a menu structure for work in general ($GUI_{menu(Work)}$) is specified, with reference to relevant design guidelines. Analyses of GUI styles also supported the development of the generic Problem element associated with the $GUI_{menu(Work)}$ - the Problem element of humans *initiating* interaction with computers - the ${}_{initiation}Element_{(Work)}$.

In general terms, a GUI$menu_{(Work)}$ is specified as: (i) a computer display which comprises, at an abstract level, a categorisation and labelling of a set of initiable interactions and, at a physical level, a menu bar, menus, menu options and labels; (ii) user manipulations which comprise, at an abstract level, selecting categories of interactions for further consideration and selecting certain interactions for initiation and, at a physical level, pointing a cursor, depressing a mouse button, dragging a cursor and releasing the mouse button; (iii) computer displays which comprise, at an abstract level, giving feedback to the user about the manipulations that the computer has detected, and at a physical level, highlighting elements of the display; and (iv) user manipulations which, at an abstract level, comprise monitoring of feedback and, at a physical level, searching the screen.

In more specific terms, GUI$menu_{(Work)}$ supports the following interaction. At a conceptual level, the interaction is as follows. The user is in control of the initiation interaction. It is assumed that, if relevant, the user has just selected domain objects to which the interaction initiated is to relate. The computer prompts for interaction category selection by displaying labels of all categories of initiable interactions, and the initiability of the interactions that these categories subsume (Smith & Mosier, 1986, 3.1.3. No's. 16 and 18[3]). Then, the user decides which interaction he or she wishes to initiate, identifies the category of interactions of which the target interaction is most likely a member, and selects this interaction category. The computer gives feedback about the selection, and the user actively monitors for this feedback. Then the computer displays the labels of the initiable interactions that the category subsumes and their initiability. The computer also displays the labels of subordinate categories of interactions. Note that only a single interaction is initiated by each menu selection (Smith & Mosier, 1986, 3.1.3. No. 2). Selection in order to view members of a category (for example, click menu name), is also distinct from selection in order to elicit feedback about an interaction to be initiated (for example, drag down) and selection in order to initiate an interaction (for example, release mouse button). Categories of interactions are also distinguished from the initiable interactions themselves (Smith & Mosier, 1986, 3.1.3. No. 6 & 31). Further, only a single level of cascading

---

[3]The numbers in this reference indicate precisely which of Smith & Mosier's guidelines the specification is thought to satisfy.

menus is permitted (Smith & Mosier, 1986, 3.1.3. No.'s 17 and 27). In addition, in the GUI$_{menu(Work)}$ specified here, initiable human-computer interactions, are differentiated according to the domain objects and attributes they seek to transform and the human and computer structures that support interaction. Assuming that users categorise interactions in a similar manner, such a differentiation ensures that a menu structure is meaningfully organised (Smith & Mosier, 1986, 3.3.3. No. 10 and 22)(see also Section 4.1.1.1) (see Figure 4.4).

Figure 4.4.: A Menu Structure for Work in General

*Domain Object/ Int've Behaviour	*Domain Object/ Int've Behaviour	*Domain Object/ Int've Behaviour
*Domain Object/ Int've Behaviour1	*Domain Object/ Int've Behaviour1	*Domain Object/ Int've Behaviour1
*Domain Object/ Int've Behaviour2	*Domain Object/ Int've Behaviour2	
*Domain Object/ Int've BehaviourN	*Domain Object/ Int've Behaviour3	
	*Domain Object/ Int've BehaviourN	

Key: * = place-holder, rather than actual menu label

If the user selects an initiable interaction, then the interaction continues as follows. The computer gives feedback about the selection. The user monitors for this feedback and the user then confirms the selected interaction for initiation. The computer gives feedback about the selection, and the user monitors for this feedback. The interaction initiated then begins. Alternatively, if the user selects a subordinate category of interactions, then the interaction continues as follows. The computer gives feedback about the selection. The user monitors for this feedback. The computer then displays the labels of initiable interactions and their initiability and the interaction continues as specified above. Note that a pure GUI$_{menu(Work)}$ does not enable menu selection to be by-passed with a keyboard command entry or menu selections to be stacked by code entry, since in such case a user is assumed to no longer experience input as direct interaction with a virtual world.

Specified at a physical level, this GUI$_{menu(Work)}$ is based on a 'pull-down' metaphor. The labels of interaction categories are displayed in one visually distinct area - a menu bar - and initiable interactions within a certain category are displayed in another visually distinct area (Smith & Mosier, 1986, 3.1.3. No. 31). Each of these areas are also distinct from other displayed information (Smith & Mosier, 1986, 3.1.3. No. 20) and sufficiently large to facilitate pointing (Smith & Mosier, 1986, 3.1.3. No. 5). Initiable interactions are displayed in a single column list format (Smith & Mosier, 1986, 3.1.3. No. 3) and groupings within a category are indicated by horizontal divisions (Smith & Mosier, 1986, 3.1.3. No. 24). The user selects a category of interactions by pointing the cursor at the category in the menu bar and 'marks' the point by depressing the mouse button (Smith & Mosier, 1986, 3.1.3. No. 4). A list of initiable interactions unfurl vertically from beneath the category label. The user points the cursor at the interaction he or she wishes to initiate and 'marks' the point. Subjectively, the user feels as if the whole interaction is executed in a smooth, continuous gesture. Feedback is given by highlighting the relevant category of interaction, or interaction by reversing the foreground and background colours, or by furling/unfurling the menu.

This GUI$_{menu(Work)}$ is conceived as a response to a *selection* of a general design Problem, specifically, a general Problem element of humans *initiating* interactions with computers to perform work effectively ($_{initiation}$Element$_{(Work)}$). Effectively, here, implies that actual contribution of initiation behaviour to overall actual performance (over all interactions) equals the required contribution of initiating behaviour to required performance[4].

---

[4]For convenience, the $_{initiation}$Element may be expressed as:

specify then implement *initiating*($\boldsymbol{U}^B$ x$\boldsymbol{C}^B$), such that $_{initiating}\boldsymbol{P}$R = $_{initiating}\boldsymbol{P}$A)

where *initiating*($\boldsymbol{U}^B$ x$\boldsymbol{C}^B$) represents interactive initiating behaviour, $_{initiating}\boldsymbol{P}$R represents the required contribution of initating behaviour to required performance and $_{initiating}\boldsymbol{P}$A represents the actual contribution of initating behaviour to actual overall performance.

4.1.1.3. Development of a GUI Menu Structure for Planning Tasks and its Associated General Problem Element

In this Section, to complete the development of the application representations, the $\text{GUI}_{menu(Work)}$ and the $_{initiation}\text{Element}_{(Work)}$ are specialised for planning tasks, with the support of the analysis of the domain of plans in the pre-conception of C^2, and guidelines for the design of menu structures.

The general Problem element of concern here - $_{initiation}\text{Element}_{(Planning)}$ - is taken to comprise human planners *initiating* planning interactions with computer-based planning-aids to produce plans effectively. It is assumed that, in principle, the implications for overall planning performance of initiating planning interactions in a certain manner may be separated from other determinants of overall planning performance and may be completely and coherently specified. In addition, it is assumed that these implications may also be differentiated according to the element of the planning domain with which they may be associated, that is, the scope, content, and view of plan objects.

Specified at a conceptual level, $\text{GUI}_{menu(Planning)}$ comprises the following interaction (see Figure 4.5). The planning-aid prompts for planning interaction category selection. The planner decides which planning interaction he or she wishes to initiate, identifies the category of interactions of which the target planning interaction is most likely a member, and then selects this planning interaction category. If the planner selects an initiable interaction, then the interaction continues as follows. The planning-aid gives feedback about the selection, and the planner actively monitors for this feedback. Then the planning-aid displays the labels of the initiable planning interactions that the category subsumes and their initiability. The planner selects an immediately initiable planning interaction. The planning-aid gives feedback about the selection, and the planner monitors for this feedback. The planner then confirms the selected planning interaction for initiation. The planning-aid gives feedback about the selection, and the planner monitors for this feedback. The planning interaction initiated then begins. Alternatively, if the planner selects a subordinate category of planning interactions, then the interaction continues as follows. The planning-aid gives feedback about the selection. The planner monitors this feedback. The planning-aid then displays the labels of initiable planning interactions and their initiability and the planning interaction continues as specified above.

Figure 4.5.: A Menu Structure for Planning Tasks

File	Objects	View	Utilities
*Plan Object1	*Domain Object1	*View Type1	*Planner specifies, Aid generates scope1
*Plan Object2	*Domain Object2	*View Type2	*Planner specifies, Aid generates scopeN
*Plan ObjectN	*Domain ObjectN	*View TypeN	*Planner specifies, Aid evaluates scope1
		Filters...	*Planner specifies, Aid evaluates scopeN
		Format...	*Planner specifies, Aid generates content 1
			*Planner specifies, Aid generates content N
			*Planner specifies, Aid evaluates content1
			*Planner specifies, Aid evaluates contentN

Key: * = place-holder, rather than actual manu label

In $GUI_{menu(Planning)}$ specified here, there are four groups of interactions that modify the scope attributes of Plan objects: (i) those in which the planning-aid generates scope according to the planner's specifications; and (ii) those in which the planner generates scope and the planning-aid prompts for the planner to state the scope that he or she has generated. (iii) those in which the planning-aid evaluates scope according to the planner's specification; and (iv) those in which the planner evaluates scope and the planning aid prompts for the planner to state his or her evaluation. There are four groups of interactions that modify the content attribute of Plan objects: (i) those in which the planning-aid generates content according to the planner's specification; (ii) those in which the planner generates content and the planning-aid prompts for the planner to state the content that he or she has generated; (iii) those in which the planning-aid evaluates content according to the planner's specification; and (iv) those in which the planner evaluates content and the planning aid prompts for the planner to state his or her evaluation. Further, there are two groups of interactions that modify the view attributes of Plan objects: (i) those which change the 'viewType' attribute; and (ii) those which change 'viewOptions' attributes, that is, 'viewContentOptions' and 'viewFormatOptions'. Finally, there is a group of interactions which identify the plan object which subsequent interactions are intended to modify. In this category, there is one interaction for each plan object (at a certain level of description).

Specified at a physical level, the GUI$_{menu(Planning)}$ is as follows. The category labelled 'Objects' comprises planning interactions in which the planning-aid is prompting the planner for input and the planner is generating or evaluating Plan scope and content. To satisfy the 'dimension' of premature commitment (Green, 1989), the planner may generate and evaluate scope and content in response to a *single* planning aid prompt. Given that scope delimits the objects for which desired states may be stated, and content specifies these states, this prompt is labelled according to the conflict domain objects implicated by the plan. In the case of military planning, for example, these objects would be objects from the domain of armed-conflict, such as Friends, Hostiles, Military Systems etc. The category labelled 'View' comprises planning interactions which change the 'view' attribute of Plan objects. The planning interaction that changes the 'viewContentOptions' attribute is labelled 'Filters...'. The planning interaction that changes the 'viewFormatOptions' attribute is labelled 'Format...'. The category labelled 'Utilities' comprises planning interactions in which the planning-aid generates or evaluates the scope and content of Plan objects according to the planner's specification. These interactions are not labelled in the GUI$_{menu(Planning)}$, since such labels are likely to reflect planning algorithms actually implemented, which are likely to vary from instance to instance, and so are best omitted from the general object. Interactions which identify the plan object which subsequent interactions are intended to modify are labelled according to the identity attribute of the plan object.

Taking the conceptual and physical specifications together, the GUI$_{menu(Planning)}$ is a specialisation of GUI$_{menu(Work)}$ specified earlier in this section. In addition, the GUI$_{menu(Planning)}$ specifies some additional aspects of the interactions, namely, the categories of interactions initiated, the labels of these categories, the range of interactions that may be initiated, and labels for selected interactions. Note, however, that GUI$_{menu(Planning)}$ still does not specify *all* aspects of the initiation interaction. Some aspects of this interaction are better specified in the instance, that is, differently for each instance of unique characteristics of the problem instance come to light.

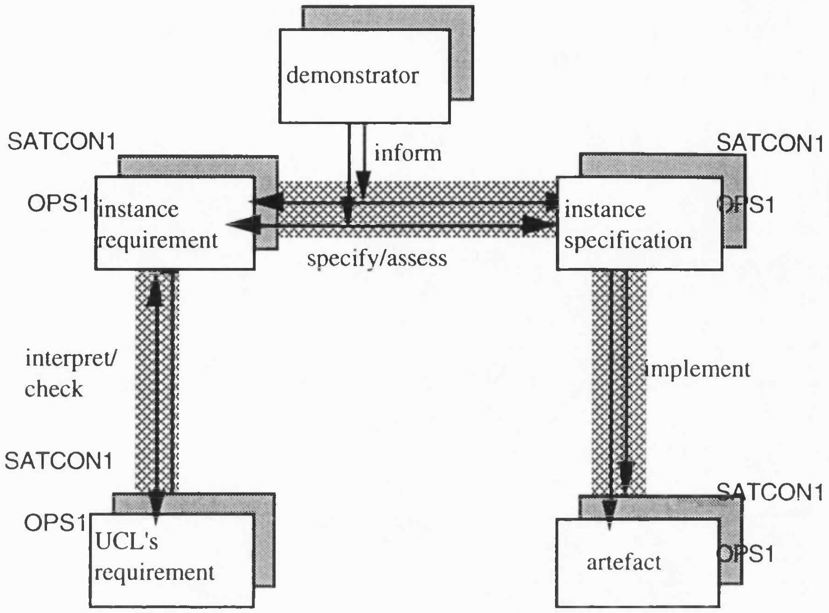
This concludes the development of the application representations to be used in this illustration of carry forward in informal Engineering.

The following section reports the design work conducted prior to executing carry forward.

4.1.2. Specification and Implementation of OPS1 and SATCON1

The design work to which the application representations are applied concerns the development of two systems reconstructed and simplified in the laboratory for the purposes of research (see Figure 4.6). These systems are, first, UCL's Off-Load Planning System, OPS1, and second, UCL's Satellite Construction System (SATCON1). For each system, client and instance requirements were presumed. Then, selected graphical user interface objects for the instance, such as plans views, and view filtering and formatting dialogues, were specified as sketches. For each system, a partial prototype was then implemented (see Figures 4.7 and 4.8). In each case, specification was supported by reviews of relevant demonstrator systems (see Acknowledgement). At this stage, each prototype comprised a stack of cards, each card simulating one type of interaction. What remained to be implemented was movement between cards, that is, initiating simulated interactions.

Figure 4.6. Design Work Undertaken



4.1.3. Re-Use and Incrementation of Application Representations in Specification

The use of the application representations in specification begins with the interpretation of UCL's client requirements for an off-load planning system and for a military satellite construction system as military planning systems implicating the general Problem of military systems (see Figure 4.9). This interpretation is performed with respect to the Problem hierarchy developed in Chapter 3. UCL's requirement for a military satellite construction system appeared to implicate a novel sub-class of design Problem - satellite construction - which exhibited the characteristics of the military planning system design Problem in general. (UCL is presumed to be one of many organisations that may express a requirement for a military satellite construction system.) The categorisation of satellite construction planning as a sub-class of military planning extended the Problem hierarchy through the addition of the new sub-class (see Figure 4.10) and prompted: (i) abstraction of a selective general requirement for military planning systems expressed as statements of required performance (see Table 4.1); and (ii) the selective instantiation of the analysis of the domain of plans (see Chapter 2) for off-load planning and satellite construction scheduling (see Table 4.2). A brief, domain-oriented analysis of the interactive planning behaviours

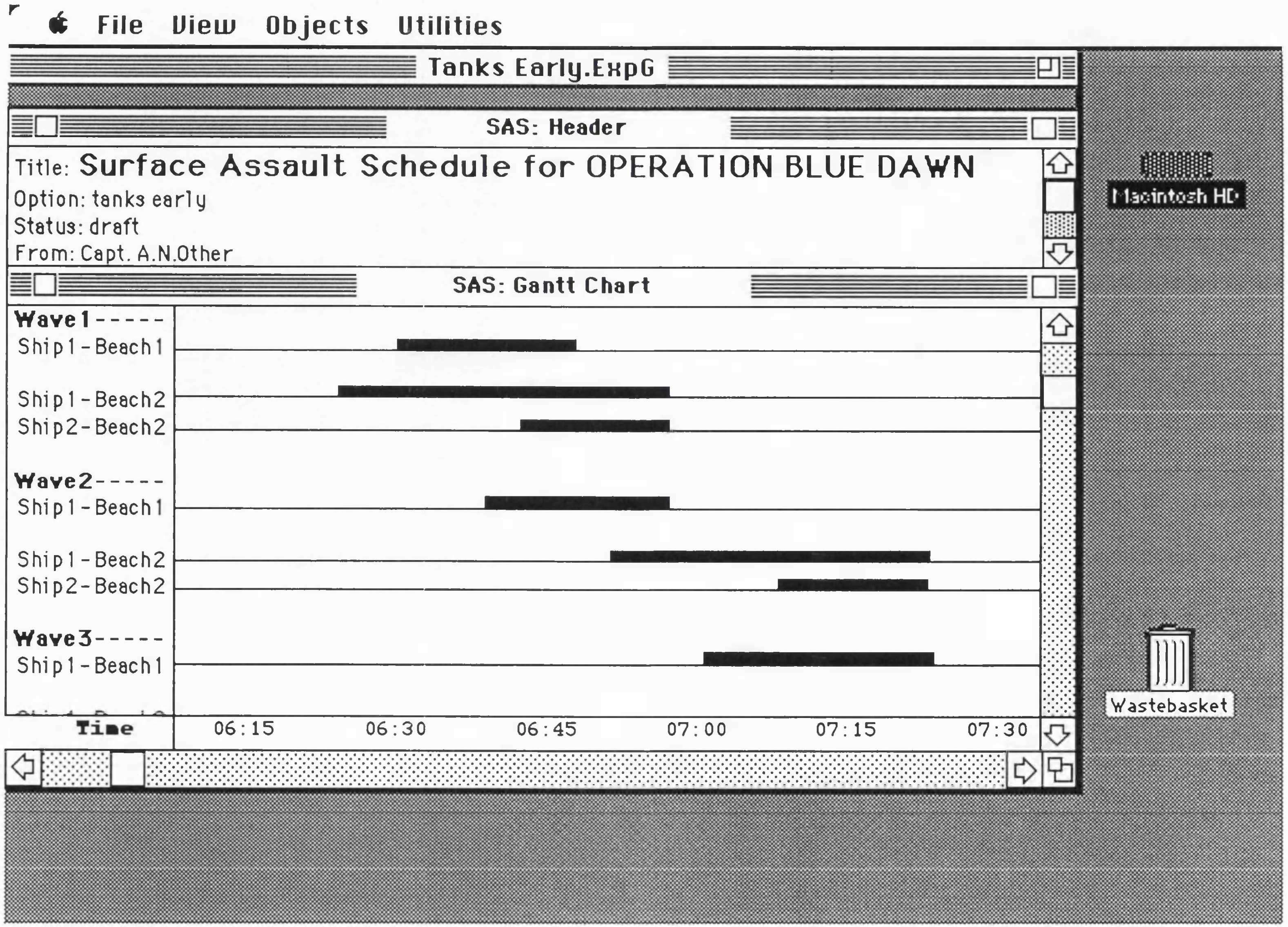


Figure 4.7: OPS1: Gantt Chart View

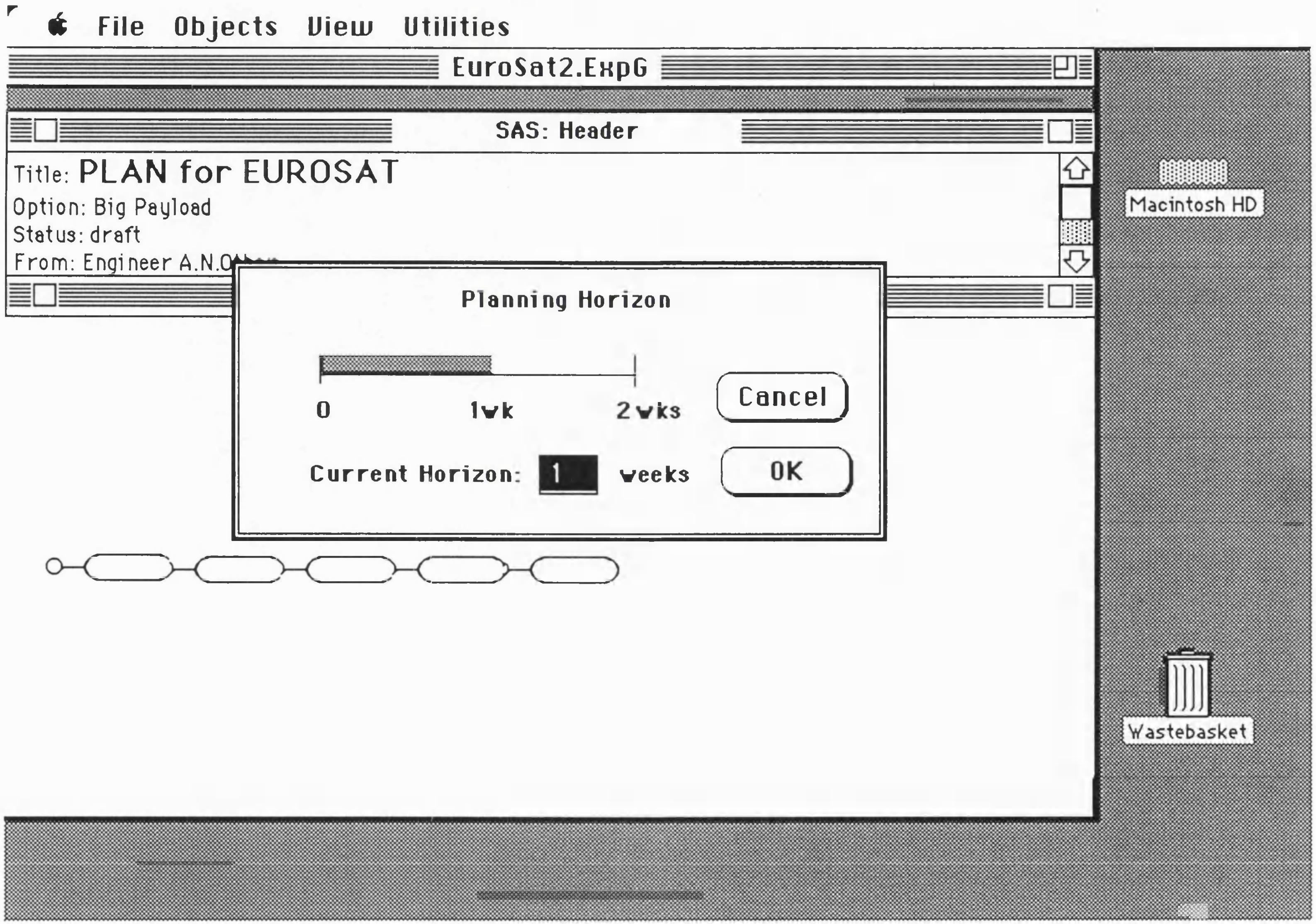


Figure 4.8: SATCON1: 'Until...' Dialogue

Figure 4.9: Use and Incrementation of Application Representations involving Problem Hierarchy, Pre-conception, General Problem Element and General GUI Objects

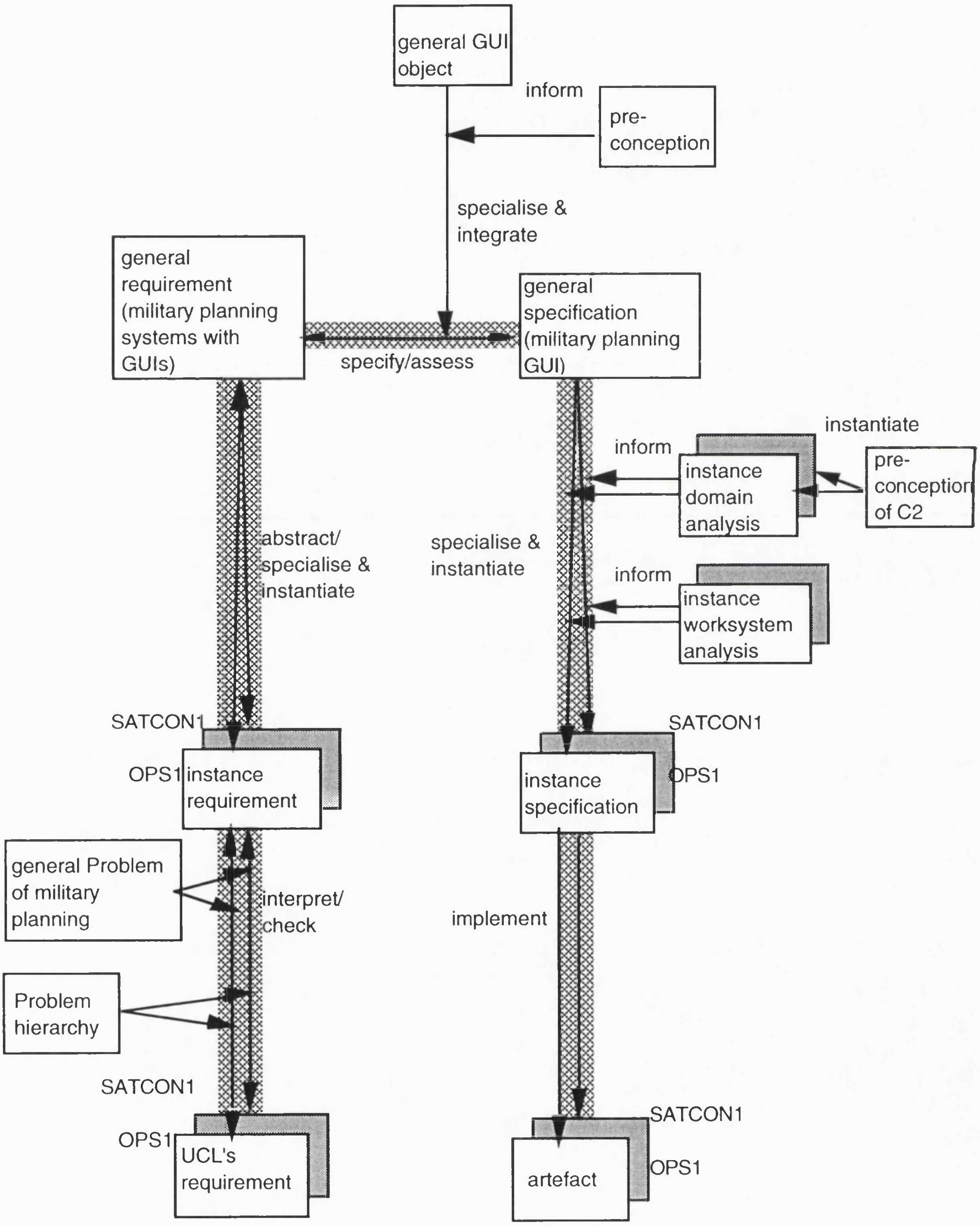


Figure 4.10: Further Incremented Problem Hierarchy

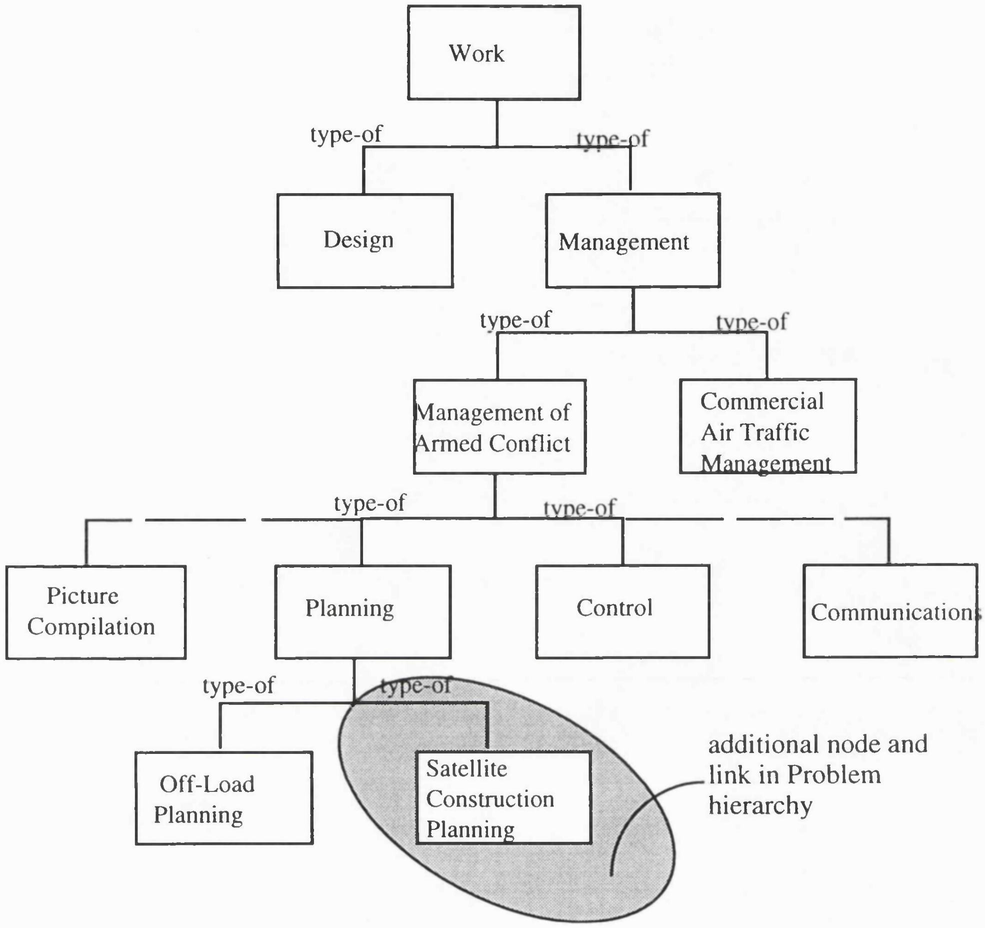


Table 4.1. A General Requirement for Military Planning Systems

---

Task Requirements

(i) future planning tasks must comprise the explicit development of more alternative plan options than current tasks. It is believed that, with more explicit options, better assessments of each option may be made, that the option finally selected will be more appropriate, and that the nature of, and rationale for, the selected option may be better communicated;

(ii) the elapsed time between the start of a future planning task and the earliest point in its time scope must be less than the current elapsed time. It is believed that delaying the on-set of planning will permit the plan to be based on higher quality information, and so the plan option finally selected will be more appropriate.

Quality Requirements

(i) the object-scope of the plan must be of higher quality (no object-scope errors);

(ii) the content of the plan must be at least as good as current plans;

(iii) content must be represented with more than the current number of view-Types, view-Content-Options and view-Format-Options. It is thought that plans may be used in many ways by many different military personnel. If plan view may be adjusted to reflect these different personnel and purposes, then a plan may better support personnel in their tasks.

Cost Requirements

(i) a planner must need less training than is currently required;

(ii) the planning-aid must process information faster than is currently processed;

Table 4.2: Selective Instantiation of Domain Analysis for Off-Load Planning and Satellite Construction Planning at UCL

Domain Object	Off-Load Planning at UCL	Satellite Construction Planning at UCL
LEVEL2 PLAN	LANDING PRIORITY TABLE	THE SCHEDULE
LEVEL3 PLAN	LOAD&STOW, SURFACE ASSAULT SCHEDULE, HELICOPTER EMPLOYMENT & ASSAULT LANDING TABLE	not distinguished
LEVEL4 PLAN	CROSS-DECKING PLAN	not distinguished
FRIENDS	LANDING FORCE	none
SYSTEMS	LANDING CRAFT, HELICOPTERS	none
MEN	none	WORKFORCE
EQUIPMENT	PONTOONS, BEACHES	COMPONENTS, STORES, VEHICLES, TOOLS

generally is also instantiated for each system (see Table 4.3). According to this analysis, the behaviour of planning worksystems may be categorised according to: (i) the plan attribute that the interaction seeks to influence; (ii) the generative or evaluative nature of the influence sought; and (iii) the type of relationship that the planner and the planning-aid exhibit during interaction. Plans are taken to possess three kinds of attributes - scope, content and view (see chapter 2). Further, planning worksystems are taken to support two kinds of relationship between planner and planning aid - first, 'prompt-provide', in which the planning-aid prompts for information and the planner generates the information sought, and, second, 'specify-implement', in which the planner specifies the computer contribution required, and the computer proceeds to make this contribution as instructed. Thus, a military planning worksystem is taken to comprise 12 categories of behaviour. These are selectively instantiated in OPS1 and SATCON1 (see Table 4.3).

Table 4.3. Selective Instantiation of an Analysis of Military Planning Worksystems Instantiated for Off-Load Planning and Military Satellite Construction Planning at UCL

Attribute Influence Relationship	**Off-Load Planning at UCL**	**Satellite Construction Planning at UCL**
Scoping Evaluative		
Prompt-Provide	none	none
Specify-Implement	'Clerical Check'	none
Scoping Generative		
Prompt-Provide	'Landing Craft...' etc.	'Components...' etc.
Specify-Implement	'Import Data'	none
Considering Content Evaluative		
Prompt-Provide	none	none
Specify-Implement	'Assess' etc.	none
Considering Content Generative		
Prompt-Provide	'Landing Craft...' etc.	'Components...' etc.
Specify-Implement	'Next Load...' etc.	'Generate...' etc.
Viewing Evaluative		
Prompt-Provide	none	none
Specify-Implement	none	none
Viewing Generative		
Prompt-Provide	'Format...', 'Filters...'	'Format...', 'Filters...'
Specify-Implement	'Gantt' etc.	'Gantt' etc.

In response to the general requirement, a general specification of a GUI interface for military planning is developed by specialising and integrating the available general GUI objects for relevant domains (see Figure 4.11). This general specification, however, is somewhat incomplete, since, at the time the work was conducted, the only available objects were: (i) the menu structure for planning tasks developed earlier in this Chapter (see Figure 4.5; and (ii) the general menu structure for personal computer file control tasks (see Figure 4.3). Selective illustration, however, was envisaged when the zero-->build strategy was devised and accepted as an necessary consequence of inadequate knowledge about relations between basic, application and design representations. A selective general specification is also adequate to illustrate carry forward and reasoning about completeness with respect to classes of specification. Specialisation of $\text{GUI}_{\textit{menu(Planning)}}$ is supported by an analysis of the domain of armed-conflict. The generic specification is then instantiated, first, for OPS1, and then for SATCON1 with the support of domain and worksystem analyses of the respective instances.

Figure 4.11.: An Integration of General GUI Objects for Military Planning Systems

File	Objects	View	Utilities
New	*Conflict Object1	*View Type1	*Planner specifies, Aid generates scope1
Open...	*Conflict Object2	*View Type2	*Planner specifies, Aid generates scopeN
Close	*Conflict ObjectN	*View TypeN	*Planner specifies, Aid evaluates scope1
Save		Filters...	*Planner specifies, Aid evaluates scopeN
Save As...		Format...	*Planner specifies, Aid generates content 1
*Specify Other Characteristics of Printed Plan			*Planner specifies, Aid generates content N
Print			*Planner specifies, Aid evaluates content1
*Military Plan Object1			*Planner specifies, Aid evaluates contentN
*Military Plan Object2			
*Military Plan ObjectN			
Quit			

Key: * = place-holder, rather than actual manu label

The specification of a menu structure for the two planning systems of concern is considered in more detail below.

4.1.3.1. Specialising a Menu Structure for OPS1

To specify a menu structure for OPS1 (GUI$_{menu(Off\text{-}LoadLab)}$), the general specification is selectively instantiated for off-load planning at UCL, with the support of domain and worksystem analyses for the instance.

In the GUI$_{menu(Off\text{-}LoadLab)}$ specified for OPS1 (see Figure 4.12), there are only three groups of interactions that modify the scope attributes of Off-Load Plan objects: (i) those in which OPS1 generates scope according to the student's specifications. There is a single interaction in this category 'Import Data', in which OPS1 imports lists of military systems participating in the operation from a Force Structure database; (ii) those in which the student generates scope and the planning-aid prompts for the planner to state the scope that he or she has generated. There are five interactions in this category, 'Landing Force...', 'Landing Craft' etc., since five types of military system may participate in off-loads - the landing force, landing craft, helicopters, pontoons and beaches[5]; and (iii) those in which OPS1 evaluates scope according to the student's specification. There is a single interaction in this group, 'Clerical Check', in which OPS1 searches for clerical errors. There are only three groups of interactions that modify the content attribute of Off-load Plan objects: (i) those in which OPS1 generates content according to the student's specification. There are three off-load planning interactions in this group, 'Next Load...', 'Next Wave...' and 'Until...', since OPS1 may generate content load by load, wave by wave or until a time defined by the student; (ii) those in which the student generates content and OPS1 prompts for the student to state the content that he or she has generated. There are three interactions in this group, 'Landing Craft...', 'Helicopters...' and 'Pontoons', since only these types of military systems may have goal states specified by the student off-load planner; (iii) those in which OPS1 evaluates content according to the student's specification. There are three interactions in this group, according to whether OPS1 provides a verbal assessment of a selected load option, a comparative verbal assessment of two load options, or a numerical comparison of all load options. There are two groups of interactions that modify the view attributes of Off-Load Plan objects: (i) those which change the 'viewType' attribute. The 'viewType' attribute of UCL's simplified off-load plans may take four values - Table, Gantt Chart, Map and Assets; and (ii) those which change 'viewOptions' attributes, that is, 'viewContentOptions' and 'viewFormatOptions'. Finally, there is a group of interactions which identify the plan object which subsequent interactions are intended to modify. In off-load planning, there are three Plan objects in addition to the Off-Load Plan - Landing Priority Table, Load and Stow and Helicopter Employment and Assault Landing Table. These options are added to the bottom of the general menu for personal computer file control.

[5]For the purposes of designing OPS1, beaches are regarded as a certain type of military system - like pontoons, only in a fixed position between the sea and the land.

Figure 4.12.: OPS1's Menu Structure

File	Objects	View	Utilities
New	Landing Force...	Table	Import Data
Open...	Landing Craft...	Gantt	Clerical Check
Close	Helicopters...	Map	Next Load
Save	Pontoons...	Assets	Next Wave
Save As...	Beaches...	Filters...	Until...
Page Setup...		Format...	Assess
Print...			Compare
LPT			All Scores
Load&Stow			
HEALT			
X-Deck			
Quit			

It is notable that $GUI_{menu(Off\text{-}LoadLab)}$ is a selective instantiation of the general specification. That is, not all the interactions specified for planning systems as a class may be instantiated through OPS1's menu structure. For example, there is no scoping interaction in which the student evaluates scope and OPS1 prompts for the student to state his or her evaluation. This interaction may not be initiated through the menu structure, and so the student has no opportunity to confirm, or disconfirm, through the menu structure, lists of assault craft participating in the operation imported from a Force Structure database. Also, no content interaction, in which the student evaluates content and OPS1 prompts for the student to state his or her evaluation, may be initiated through the menu structure. That is, the student has no opportunity through the menu structure to 'cross-off' unacceptable options that OPS1 has generated. Thus, OPS1 as a planning system is an incomplete instance of the general specification, since such behaviours are assumed to be unnecessary to satisfy UCL's presumed requirement.

4.1.3.2. Specialising a Menu Structure for SATCON1

In the menu structure specified for SATCON1 ($GUI_{menu(SatConLab)}$) (see Figure 4.13), there is only a single interaction that modifies the scope attributes of a satellite construction schedule - the one in which the student generates scope and SATCON1 prompts for the student to state the scope that he or she has generated. There are five interactions in this category 'Components...', 'Stores...' etc., since five types of military system are involved in satellite construction - components of the satellite, stores, workforce, vehicles (for transporting components around the construction area) and construction tools. There is only a single interaction that modifies the content attribute of the Satellite Construction Schedule - the one in which SATCON1 generates content according to the student's specification. There are three satellite construction scheduling interactions in this group 'Generate', 'Cycle by Cycle' and 'Ask for Info', since SATCON1 may generate a complete schedule automatically, one cycle at a time, or until additional information is required from the student. There are two groups of interactions that modify the view attributes of the satellite construction schedule: (i) those which change the 'viewType' attribute. The 'viewType' attribute of UCL's simplified satellite construction schedules may take three values - a text list, a gantt chart, a 3-D view of the satellite (in some stage of completion) and resource utilisation; and (ii) those which change 'viewOptions' attributes, that is, 'viewContentOptions' and 'viewFormatOptions'.

Figure 4.13 : SATCON1's Menu Structure

File	Objects	View	Utilities
New	Components...	Plan	Generate
Open...	Stores...	Schedule	Cycle by Cycle
Close	Workforce...	World	Ask for Info
Save	Vehicles...	Resource Utilisation	
Save As...	Tools...	Filters...	
Page Setup...		Format...	
Print...			
Quit			

It is notable that GUI*menu(SatConLab)* is a selective instantiation of the general specification. That is, not all the interactions specified for planning systems as a class may be initiated through SATCON1's menu structure. For example, there is no scoping interaction in which the student evaluates scope and SATCON1 prompts for the student to state his or her evaluation, there is no content interaction in which the student evaluates content and SATCON1 prompts for the student to state his or her evaluation. Finally, there is only a single plan object in satellite construction scheduling, so the plan object intended to be modified by subsequent interactions may be presumed and another may not be selected.

Following instantiation of the general specification for OPS1 and SATCON1, the implementation of the prototypes was completed.

This section concludes the illustration of carry forward in a manner characteristic of informal HCI Engineering in specification. The following section briefly illustrates carry forward in a manner characteristic of mature Craft.

4.2. Explicit Craft Specification

This section briefly illustrates the conventional explicit craft specification which, it is assumed, would have been conducted, had there been no informal Engineering alternative.

4.2.1. Application Representations

The application representations that support the equivalent Craft evaluation of OPS1 are, first, heuristics for menu design, and second, specifications of the menu structure of existing military planning systems. Both application representations were obtained from the human factors literature.

Heuristics

A design heuristic, here, (see section 1.1.3.2.) is taken to be a rule of thumb which was acquired by reflecting upon, and then abstracting from, design experience, and which consequently has the status of an informed, and shared opinion. Collections of heuristics may be organised according to the range of design problems to which they apply and the perspective from which they are formulated.

This illustration applies a selection of two sets of heuristics[6] thought to apply to interaction generally, the first (Green, 1989) formulated from a cognitive perspective, the second from a usability perspective (Denley et al, 1993). The illustration also applies a selection of two sets of heuristics thought to apply to more specific issues, command language design and menu design, and formulated from a usability perspective (Smith & Mosier, 1986, 3.1.3. and 3.1.5).

<u>General Heuristics, from a cognitive perspective</u> (after Green, 1989)

• hidden dependencies: if users prefer to think opportunistically, and if dependencies exist in their work, try to ensure that such dependencies may be easily perceived from all directions by, for example, providing referencing and browsing tools;

• premature commitment: if users prefer to think opportunistically, try to let users perform actions in any order. Converting the mental generative

[6]In actual fact, the derivation of the 'heuristics' cited is, in all cases, mixed and/or unclear.

order into an acceptable instruction order by look-ahead (that is, working it all out in the mind before starting the sequence in which all actions will be performed) may impose a high workload.

General Heuristics, from a usability perspective (Denley et al., 1993)

• compatibility: users should be able to apply knowledge they have gained from outside the system. This will help the user in learning to work with the system and reduce the amount of new information to be interpreted and remembered (e.g. cognitive load, memory load)

• coherence: users should be able to generalise their experience between system components. This will help the user in learning to work with unfamiliar system components and reduces memory load.

• simplicity: the number and complexity of necessary user actions should be reduced to a minimum. Thus, cognitive costs (e.g. memory load) and physical costs (e.g. keystrokes) are minimised.

• completeness: information about tasks which can be performed, as well as about states of the machine, should be complete in such a way that all options are presented simultaneously. This should reduce memory load of the user and thus give them the opportunity to decide between all potential options and task actions.

• support orientation: if the information to be presented is too complex or covers more than is possible to present at one time, the user should be helped to find the relevant information by giving them support orientation. This enables the user to easily find the relevant information and helps them to perform even complex tasks.

Specific Heuristics, Menu Structures (Smith & Mosier, 1986, 3.1.3.)

• complete display of menu options: design a menu to display all options appropriate to any transaction (except a familiar set of general control options);

• consistent display of menu options: when menus are provided in different displays, design them so that option lists are consistent in wording and ordering;

• hierarchic menus for sequential selection: when menu selection must be made from a long list, and not all options can be displayed at once, provide a hierarchic sequence of menu selections rather than one long, multipage menu (except when a long list is structured for other purposes, such as a parts inventory);

• provide a general menu of basic options as the top level in a hierarchic menu structure, a 'home base' to which a user can always return as a consistent starting point for control entries;

• minimal steps in sequential menu selection: when users must step through a sequence of menus to make a selection, design the hierarchic menu structure to minimise the number of steps required;

• logical ordering of menu options: list displayed menu options in a logical order; if no logical structure is apparent, then display the options in order of their expected frequency of use, with the most frequent listed first;

• logical grouping of menu options: format a menu to indicate logically related groups of options rather than as an undifferentiated string of alternatives

Specific Heuristics, Command Language (Smith & Mosier, 1986, 3.1.5.)
• functional wording: design a command language so that a user can enter commands in terms of functions desired, without concern for internal computer data processing, storage and retrieval mechanisms;

• familiar wording: choose words for a command language that reflect the user's point of view, and correspond to the user's operational language;

• distinctive meaning for commands: design words in a command language so that they are distinctive from one another, and emphasise significant differences in function.

Menu structures of Existing Military Planning Systems
BATTLE is a planning system for supporting arms target allocation (Slagle & Hamburger, 1985). The top-level of this menu structure groups

behaviours intended to characterise weapons and targets ('Alter' menu), behaviours intended to modify how the computer should go about planning ('Constraints' menu), and behaviours intended to produce the plan ('System' menu).

4.2.2. Re-Use and Incrementation of Application Representations

Initial Specification of OPS1's menu structure

Since BATTLE and OPS1 are both military planning systems, an initial specification for a menu structure for OPS1 may similarly group modifications to participants, user interventions in the computer's planning behaviour, and producing a plan (see Figure 4.14).

Figure 4.14.: Iterative Specification of a Menu Structure for OPS1

Initial Specification

Participants	Constraints	System
Add New Unit...	Remove Unit from Consideration...	Devise a Plan
Change Name of Unit...	Prescribe an Allocation of Units...	Print Help...
Modify type/speed/ capacity of Unit...	Prevent an Allocation of Units...	Print Commands
Print List of Units	Undo a Prescription or Prevention...	Halt Program Execution

Subsequent Specification

File	Assets	View	Utilities	Plans
New	Landing Force...	Table	Import Data	LPT
Open...	Landing Craft...	Gantt	Clerical Check	Load&Stow
Close	Helicopters...	Map	Set Beaches...	HEALT
Save	Pontoons...	Assets	Next Load	X-Deck
Save As...		Filters...	Next Wave	
Page Setup...		Format...	Until...	
Print...			Assess	
Quit			Compare	
			All Scores	

Iteration

This initial specification may be immediately assessed using an 'heuristic evaluation' technique similar to that suggested by Nielsen (1989). In such assessments, a specification may be assessed with respect to the criterion implicit in the heuristic. If the specification fails to satisfy a criterion, then the recommendation suggested by the heuristic may be incorporated in subsequent specifications. An iteration in the specification of the menu structure for OPS1 based upon heuristic assessment follows:

(i) inconsistent display of menu options: weapons and targets are sometimes labelled 'participants' and sometimes 'units'. OPS1's menu should use the familiar, off-load term 'Assets' throughout;
(ii) illogical grouping of menu options: print options are dispersed throughout the menu structure. OPS1's menu structure should group 'Print' commands functionally and in a manner compatible with off-load planners' experience of other, software packages, that is, together and with other 'File' commands, as per interface style 'File' menu object. Further, since off-load planning involves a number of distinct plans, options for accessing these plans should also be grouped together under a 'Plans' menu;
(iii) premature commitment/meaningfulness of names: when defining the assets involved in an off-load, planners may be assumed to organise this information around a 'list of participants'. However, the initial specification requires off-load planners to decide how they wish to modify this list, before selecting the asset they wish to modify. Consequently, off-load planners searching for menu option names that display assets to be modified may find menu option names that display potential modifications to lack meaning. Premature commitment of this kind may increase the off-load planner's workload. Thus, the options that concern the definition of participants should reflect types of asset, that is, 'Landing Craft...', 'Helicopters etc.';
(iv) simplicity/minimal steps: to modify the type, speed and carrying capacity of assault craft, an off-load planner must currently select the 'Modify type/speed/capacity' option and then select 'Capacity' from the sub-menu. To display a 'flatter' menu structure, the planner should simply select the type of asset he or she wishes to define, and then declare its type/speed etc.;
(v) completeness: BATTLE presented the user with only a single view of the weapon allocation plan - a map. OPS1, in contrast, provides a number of alternative views. The off-load planner should be aware of, and have immediate access to, these options.

The output of this iteration is a menu shown in Figure 4.14 (see Figure 4.4). Note that the explicit Craft menu structure uses the off-load term 'Assets' rather than the more general term 'objects', and so groups the option 'Set Beaches...' under the 'Utilities' menu, since, to an off-load planner, a beach is not an asset. Further, options for 'going to' related plans are grouped under a 'Plans' menu. Although this grouping is 'logical' it is not, in fact, coherent with other systems with which the user may be familiar (see heuristics). Subsequent user trials and further iteration would be expected to identify and remedy this weakness.

Specification of SATCON1's menu structure

Since BATTLE and SATCON1 are both military planning systems, an initial specification for a menu structure for SATCON1 may similarly group modifications to participants, user interventions in the computer's planning behaviour, and producing a plan (see Figure 4.15).

Figure 4.15.: Iterative Specification of a Menu Structure for SATCON1

Initial Specification

Participants	Constraints	System
Add New Unit...	Remove Unit from Consideration...	Devise a Plan
Change Name of Unit...	Prescribe an Allocation of Units...	Print Help...
Modify type/speed/ capacity of Unit...	Prevent an Allocation of Units...	Print Commands
Print List of Units	Undo a Prescription or Prevention...	Halt Program Execution

Subsequent Specification

File	Constraints	Planning
New	Components...	Generate...
Open...	Stores...	Plan
Close	Workforce...	Schedule
Save	Vehicles...	World
Save As...	Tools...	Resource Utilisation
Page Setup...		Filters...
Print...		Format...
Quit		

Iteration

This initial specification of SATCON1's menu structure is again refined using heuristic evaluation. In the context of SATCON1, however, the heuristics are interpreted differently. Consequently, evaluation of SATCON1's initial menu structure results in a menu structure for SATCON1 that categorises, labels and displays initiable behaviours differently to OPS1's menu structure (see Figure 4.15). For example, in this instance, having consulted end-users, it may be reasoned that it is 'logical' to group menu options according to whether they concern setting 'constraints' or manipulating the 'plan' itself. In addition, note that a single menu option 'Generate...' is used to initiate the three interactions in which the planning-aid generates the plan according to the users specification - a sub-dialogue box pops-up following selection of the 'Generate..' options, and the users then selects which of the three generation interactions he or she wishes to conduct. Thus, although this menu structure is grouped 'logically', the display of the options is not 'complete' and the number of steps in option selection is not 'minimal' (see heuristics). Subsequent user trials and further iteration would be expected to identify and remedy this weakness.

OPS1 and SATCON1 would then be implemented. This description concludes the illustration of explicit craft specification.

4.3. Assessment of Case Report

To summarise the outcome of different manners of carry forward in specification, the informal Engineering and explicit Craft alternatives are shown in Figure 4.16.

4.3.1. Scope and Content of Specifications

To consider the scope and content of the generic menu structures, the instantiations of the generic design for OPS1 and SATCON1 were discussed with individuals involved in the development of actual demonstrator systems for off-load planning and satellite construction (see Acknowledgement). The prototypes supported the discussions. The discussions sought to identify potential errors and difficulties that a human planner attempting to produce a plan may encounter when using the selective instantiation of the general specification. (Other user roles, such as plan user, or domain modeller were not considered). In each discussion, the author demonstrated each menu option in turn, and the developer raised issues of concern to him or her.

In essence, the potential for ineffectiveness suggested by the developers were attributed by the author to the requirements presumed or the limitations of the pre-conception. Both the presumption of requirements and the use of the pre-conception is necessitated by the zero-->build research strategy.

Figure 4.16: A Summary of Alternative Manners of Carry Forward in Specification: the Case of Menu Structures for OPS1 and SATCON1

Informal Engineering

OPS1

File	Objects	View	Utilities
New	Landing Force...	Table	Import Data
Open..	Landing Craft...	Gantt	Clerical Check
Close	Helicopters...	Map	Next Load
Save	Pontoons...	Assets	Next Wave
Save As.	Beaches...	Filters...	Until...
Page Setup		Format...	Assess
Print			Compare
LPT			All Scores
Load&Stow			
HEALT			
X-Deck			
Quit			

SATCON1

File	Objects	View	Utilities
New	Components...	Plan	Generate
Open...	Stores...	Schedule	Cycle by Cycle
Close	Workforce...	World	Ask for Info
Save	Vehicles	Resource	
Save As..	Tools...	Utilisation	
Page Setup			
Print			
Quit			

Explicit Craft

OPS1

File	Assets	View	Utilities	Plans
New	Landing Force..	Table	Import Data	LPT
Open..	Landing Craft..	Gantt	Set Beaches	Load&Stow
Close	Helicopters...	Map	Clerical Check	HEALT
Save	Pontoons...	Assets	Next Load	X-Deck
Save As...		Filters...	Next Wave	
Page Setup		Format..	Until...	
Print			Assess	
Quit			Compare	
			All Scores	

SATCON1

File	Constraints	Planning
New	Components...	Generate
Open...	Stores...	Plan
Close	Workforce...	Schedule
Save	Vehicles...	World
Save As	Tools...	Resource
Page Setup		Utilisation
Print...		Filters...
Quit		Format...

For example, it may be suggested that planning with the actual demonstrators requires more constraints to be set than appeared to be possible in the prototype. A constraint, here, is a work goal that is not the ultimate objective, but nevertheless a criteria for evaluating how well planning has been performed. For example, in off-load planning, the number of landing craft that can land on a beach at any one time constrains possible plans, but the prototype provided no obvious opportunity to declare such information. The 'Beaches' option of the

'Objects' menu appeared most relevant, but the prototype appeared to simply concern the identity of the beaches involved. However, this suggestion may be rebutted by stating that the prototype is only an initial version, that the dialogues have been implemented as required given the requirements presumed by the research, and that menu design was supported by a domain analysis instantiated from the pre-conception. In the example given, the interaction initiated by the 'Beaches...' option is specified as 'the off-load planning-aid prompts for input, and the off-load planner generates and evaluates the scope and content of the off-load plan'. So, in principle, an off-load planner should be able to state the number of landing craft that can land on a beach at any one time under this option if required. The dialogue implemented does not explicitly provide a facility to declare such information because such a requirement was not presumed by the research, and because the domain analysis conducted identified only domain objects and their attributes, and not their relations (and not because, in principle, such an interaction may not be initiated through an instantiation of a generic GUI object).

As another example, it may be suggested that planners may find the prototype's grouping of functions disorderly and illogical. For example, the components of a satellite to be integrated, the order in which they are to be integrated, and other task goals and constraints, may be declared during 'domain modelling'. This activity typically occurs well in advance of planning, rather than as the plan is generated. But in the prototype, scoping the plan appears to occur in close conjunction with the generation of the plan, and domain modelling does not seem to have been considered. This suggestion may also be rebutted by stating that the prototype is only an initial version, that the dialogues have been implemented according to the requirements presumed by the research, and that menu design was supported by a domain analysis instantiated from the pre-conception. In particular, the menu structure of the prototype is an instantiation of a generic menu structure for planning tasks. As such, and as a consequence, the menu structure does not initiate interactions other than those which are intended to change the state of a plan. In the example given, 'domain modelling' interactions may well be necessitated by the requirements to which the developer of the actual demonstrator is responding. The menu structure implemented does not explicitly support initiation of 'domain modelling' interactions, because the requirement to do so was not presumed by the research, and because the domain analysis concerns the domain of plans, (and not because, in principle, the domain of 'domain modelling' could not be analysed, because a generic menu structure for initiating 'domain modelling' interactions could not be specified, or because a menu structure for domain modelling could not be integrated with a menu structure for planning in a general specification.

Thus, the menu structures instantiated for the hypothetical prototypes are sufficient to rebut unfavourable analytic assessment by the developers of equivalent, actual demonstrator systems. One has failed to establish that the generic menu structures are ineffective. However, it must be recognised that analytic assessment of one fragment of an interface in the absence of the complete artefact is an unreliable method of assessment. That said, such an assessment method is all that is permitted within the zero-->build strategy.

It is notable that the instantiations of the generic menus are more consistent, both in terms of their categorisation of interactions and their labelling, than their bespoke equivalents. Such consistency follows from the specification of these menus through selective instantiation. For example, the instantiated menu structures for both OPS1 and SATCON1 comprise 'Objects', 'View' and 'Utilities' menus, and each menu comprises options that initiate interactions of the same type. Some of these options, such as 'Filters...' and 'Format...' also have identical labels. Consistency is widely recognised as a characteristic of user interfaces that greatly enhance usability. Specifically, users who operate more than one military planning system would be expected to find the consistent menu structures more learnable and feel more confident that they could use the system for their purposes (Kellog, 1987; Barnard et al., 1981). Given that UCL's requirements are presumed, however, it is not clear how many users are of this type.

It is also notable that the explicitly crafted menu structures possessed minor design flaws, specifically, inconsistent grouping and incomplete display, of menu options. It was expected that these flaws would be identified and remedied during the next iteration in development. However, as noted in chapter 1., if such iterations require the delivery of an artefact to be delayed, then the opportunity to conduct such iterations may be denied, or provided at the cost of permitting the project to 'slip'. Thus, as is too often the case in explicit Craft, the menu structures for OPS1 and SATCON1 are, in some respects, unfinished.

4.3.2. Reasoning About Completeness/Selectivity

The informal Engineering specification is different from the explicit Craft specification in terms of how the specification was conceived and so how the completeness and/or selectivity of the specification was reasoned about. The informal Engineering specification conceived the interface to be specified as an instance of a class (see section 1.1.4). Consequently, claims about the completeness and/or selectivity of the instance may be reasoned about with reference to the class. For example, the class specification indicated that four groups of interactions modify the scope attributes of Plan objects: (i) planning aid generates scope according to the user's specifications; (ii) user generates scope and the planning-aid prompts for the planner to state the scope that he or she has generated; (iii) planning aid evaluates scope according to the user's specification; and (iv) user evaluates scope and the planning aid prompts for the user to state his or her evaluation. However, the specifications for the instances indicated that only three of these groups were required for OPS1 - OPS1 generates scope according to the student's specifications ('Import Data...'), student generates scope and OPS1 prompts for the student to state the scope that he or she has generated ('Beaches...' dialogue etc.), and OPS1 evaluates scope according to the student's specification ('Clerical Check') - and only one of these groups was required for SATCON1 - student generates scope and SATCON1 prompts for the student to state the scope that he or she has generated ('Components...' etc.). Thus, both OPS1 and SATCON1 may be regarded as selections of the general specification, but OPS1 may be regarded as a more complete instantiation than SATCON1.

In contrast, the explicit Craft evaluation addressed OPS1 as a problem instance related to other instances in poorly specified ways (see section 1.1.4). Also, the selectivity of the menu structures used in previous planning systems is poorly specified. Since the previous and forth-coming problems are not fully conceptualised, it is difficult to be explicit how complete and/or selective the forth-coming specifications are likely to be on the basis of previous work. For example, OPS1 and BATTLE are both conceived as military planning systems, but, in the illustration of explicit Craft, the nature of 'military planning' is not developed. Consequently, the completeness and/or selectivity of the specification may not be reasoned about, or worked towards beyond consideration of its adequacy for the artefact under development. The Craft specification encourages completeness and appropriate selectivity by reasoning with heuristics formulated within multiple perspectives, in order to increase the number of design options likely to be considered. The more design options considered, it is thought, the more appropriate the options selected are likely to be for the forth-coming instance. For example, some heuristics are formulated from a cognitive perspective, and some from a usability perspective. Heuristics may also be interpreted differently, depending upon the perspective of the reader. For example, the heuristic concerning premature commitment may, or may not be perceived as relevant to the issues of menu names, depending upon whether the reader adopts a limited or liberal view of the underlying argument about opportunistic planning. Adopting a cognitive perspective and a liberal view of opportunistic planning enables consideration of the option of 'grouping menu options according to type of asset', an option that, in this illustration, was not suggested within a human factors perspective.

It is important to note that reasoning about the completeness and/or selectivity of specifications with respect to classes of specification is an *addition* to reasoning about completeness and/or selectivity using multiple perspectives (see section 1.1.3). In principle, a general Problem may be conceived from a number of different perspectives and a number of general specifications could be developed. In this illustration, the Problem was conceived from a single, domain-oriented, systems perspective to encourage completeness. A single such perspective was adopted, because the class approach to reasoning about completeness and/or selectivity fails if the class specification is, itself, incomplete, and a domain-oriented, systems perspective seems to encourage complete specification by integrating human factors, software engineering and end-user concerns. However, had resources permitted, this illustration could have adopted additional perspectives and reasoned about the completeness and/or selectivity of instance specifications using both the class approach and the multiple perspective approach.

4.3.3. Adequacy of Illustration of Informal Engineering

Consideration of how well this report illustrates carry forward in informal HCI Engineering suggests that the deficiencies of the initial attempt have been remedied (see section 3.3). Re-use was fully carried through - the pre-conception of the domain of C^2 supported, first, the instantiation of an analysis for off-load plans, and second, the instantiation of an analysis for satellite construction schedules (see section 1.1.2). Further, design work involved the development and instantiation of a class specification - a GUI for military planning systems (see section 1.1.4). In addition, and as in the illustration of informal Engineering evaluation, the additional means of reasoning about the completeness of development products provided by informal Engineering was used, an informal *class* of Problem was addressed and a knowledge representation - the Problem hierarchy - was incremented. Thus, the additional opportunities for carry forward were, in this illustration, created and exploited (see section 1.1.3). However, the general specification of a GUI for military planning systems was somewhat rudimentary, since such a specification was produced by specialising and integrating relevant generic GUI objects, and, at the time of writing, only two such objects were available - the menu structure for personal computer file control tasks and planning tasks. Since that time, however, additional relevant generic object specifications have been produced, specifically, a number of generic plan views by Van Putten (1993), and so, with time, the general specification may be expected to become increasingly complete. Another weakness of the illustration is that the client, instance and general requirements were presumed and so remained implicit. Explicit statement of these requirements, however, is necessary to evaluate the effectiveness of the general and instance specifications and the final artefact. Thus, although the illustration exposes aspects of the case study that are characteristic of informal Engineering, it fails to reveal all aspects of the work that are necessary to consider the effectiveness in practice of the form of HCI realised. Such limitations are, however, inevitable given zero-->build (see section 1.2.1).

4.3.4. Relative Effectiveness of Informal Engineering Specification in Practice

Overall, judgements about the relative effectiveness in practice of the informal Engineering and explicit Craft specifications rest upon: (i) the perceived value of the savings of time and effort achieved by reasoning about completeness with respect to class specifications, rather than selectively re-using other instance specifications and interpreting heuristics for the instance at hand; (ii) the perceived value of the additional investment required to develop the pre-conception, Problem hierarchy, Problem expressions, general Problem elements and generic GUI object specifications, rather than documenting development work and developing design heuristics; (iii) the perceived value of additional opportunities to apply knowledge to development work; (iv) the perceived value of class specifications, that is, the design of an infinite number of instances, rather than the instance at hand; and (v) the perceived value of consistent user interfaces to military planning systems (see section 1.2.1). In this case, and in the author's opinion, if *complete and coherent* class specifications may be developed in response to *actual* requirements, then the cost of developing the pre-conception etc. may be justified by the additional opportunities to apply knowledge, the ease of reasoning about completeness, and the consistency of designs. That is, if scaled-up and transferred to 'real world' problems rather than reconstructed ones, the value of carry forward in informal Engineering may warrant the investment of its development. The reason for this judgement is that, in this case, the design of the generic menu structure was a difficult, time-consuming activity, which required considerable fine-tuning and refinement over a period of weeks/months. Instantiation of the generic specification, in contrast, was a relatively simple and straight-forward activity. For example, SATCON1's menu structure was developed in hours/days following brief review of the actual demonstrator system and its documentation.

Such a judgement rests upon a number of important assumptions. First, it is assumed that military planning work is allocated such that a single user does indeed use a number of different planning systems and so may indeed benefit from consistency across instances. Second, it is assumed that developing and applying the generic menu structure was approximately as difficult as developing and applying the other generic GUI objects necessary to develop complete and coherent class specifications in response to actual requirements. It is difficult to form an opinion about this assumption at this point in time.

The next step in the progressive evolution of informal HCI Engineering, it is suggested, should seek to develop complete and coherent general requirements and complete and coherent general specifications for a class of actual instances. Such work would require the development of additional generic GUI objects and the full specification of the desired form of HCI prior to its realisation. Given the contribution of this thesis, such work may be possible.

This section concludes the second attempt to realise carry forward in a manner characteristic of informal HCI Engineering. The following chapter briefly reviews progress made.

Chapter 5: Closing Remarks

5.1. Response to the Conception Papers

When reviewing this thesis as a whole, it may be enlightening to consider its outputs as a response to the two papers which provided crucial background - Dowell & Long's conception of HCI as Engineering, and Long & Dowell's alternative conceptions of HCI (Dowell & Long, 1989; Long & Dowell, 1989) (see also chapter 1). Taken together, the conception papers suggest that the most effective form for the discipline of HCI is, in principle, that of formal Engineering. Formal Engineering knowledge would be operationalisable, generalisable and testable. It would also be prescriptive, and may take the form of formal Engineering principles. Formal Engineering practices would comprise specification and implementation and the Problems addressed would be 'hard'. In contrast, the outputs of this thesis are claimed to be as follows:

(i) an extended analysis of alternative forms of HCI, with particular reference to carry forward and a critique of the background conception papers;

(ii) a pre-conception of the domain of C^2;

(iii) an initial, and in some respects unsatisfactory, case history of carry forward in late, summative evaluation;

(iv) a subsequent, and in some respects more satisfactory, case history of carry forward in specification;

(v) assessments of the effectiveness in practice of carry forward in a manner characteristic of informal Engineering, relative to the next best alternative (taken to be explicit Craft).

Since this thesis has not developed formal Engineering principles, nor sought to, it has self-evidently not progressed HCI in the way envisaged by the conception papers. Equally evidently, however, this thesis *has*, in common with the conception papers, sought to progress the discipline and to progress it towards Engineering. This thesis has also conceptualised the design Problems that it has addressed in terms of the (informal) concepts offered by the conception papers. Thus, a relationship between this thesis and the conception papers is self-evident, but the nature of this relationship is less clear. This relationship will be considered in this section.

In attempting to contribute to the enterprise of progressing HCI towards Engineering, this thesis encountered a number of difficulties with the background papers. These difficulties concerned:

(i) comprehension of the objective[1];

(ii) the strategy for achieving this objective;

(iii) the risks associated with attempts to achieve this objective.

First, it was difficult to comprehend the objective for the reader that was suggested by the background conception papers. In the conception papers, the specification of formal Engineering as a form of the discipline of HCI is brief, and there are no examples of the kind of knowledge, practices and problems that this form of the discipline is envisaged to comprise. The background papers present only a set of related concepts for expressing the general design Problem more formally, and which might be embodied in formal Engineering principles (Dowell & Long, p. 1513 and p. 1521). Formal Engineering principles are the focus of the objective suggested to the reader. However, from its description, a formal Engineering principle appears to be very unlike any HCI knowledge currently available, and principles are contrasted with HCI knowledge that *is* currently available, such as heuristics, so it is difficult to formulate a more concrete description of a principle. In fairness, of course, if a formal Engineering principle were already available, then there would be no need to suggest the development of more advanced forms of HCI - its most effective form would already have been achieved. However, without the opportunity to comprehend an objective by being shown (ostension), or specification in detail, it is difficult for a reader to comprehend the objective suggested, let alone assess it.

Second, it was difficult to envisage how progress towards the objective may be made. The set of concepts provided in the conception papers are introduced as a minimal starting point, a "pre-requisite for formulation of [formal] engineering principles", a "speculative" and "unvalidated" set of concepts which "continues to be developed in support of, and supported by, the research of the authors", and whose "power lies in the completeness and coherence of its definition of concepts" (p. 1520 and 1521). However, beyond suggesting these concepts as an initial step towards Engineering, no research strategy - for the further development of the conceptions, the

[1] the objective for the reader suggested by the conception papers.

development of formal principles, or the development of knowledge of any other kind - is suggested. In fairness, of course, an objective is not invalidated by the absence of a strategy. However, without such a strategy, it is difficult for a reader to give the development of formal Engineering principles the "serious consideration" the authors seek (p. 1533).

Third, it was difficult to deal with the risk associated with attempts to develop formal Engineering principles, specifically, the risk of conducting the "significant research program" that Dowell & Long envisage, but failing to produce a formal Engineering principle. Dowell & Long imply that such risks are acceptable, since the need to progress HCI is great and the determinacy of HCI's design concerns remains to be established. Current practice, they suggest, is poorly integrated into systems development (which nullifies any influence it might exert), of suspect efficacy, and inefficient, and shows insufficient signs of systematic and intentional progress (p. 1516). Human-computer interactions, they suggest, are clearly to some useful degree deterministic, so the current absence of formal Engineering knowledge may be symptomatic of the early stage of the discipline's development, rather than a reflection of the inherent indeterminacy of human-computer interactions (p. 1533). This thesis, in contrast, highlighted another implication of Dowell & Long's position - that since the determinism of human-computer interactions remains to be independently established, such interactions are *not necessarily* deterministic enough to support the development of formal Engineering principles (although they might be). In the absence of example formal Engineering principles, or a more complete view of the 'softness/hardness' of design Problems, it is difficult to identify interactions which are likely to be *sufficiently* determinate. Consequently, the risk of conducting research but failing to produce a principle is difficult to estimate. However, relative to the risk associated with the development of conventional forms of discipline knowledge for innovative kinds of work or Technology, for example, 'heuristics for the design of interactive, educational television', the risk associated with the development of formal principles may be assumed to be at least considerable. Further, to this 'considerable risk' must be added the risks associated with pursuing objectives which are difficult to comprehend, and for which there is no research strategy. It is difficult for a reader (other than reckless one) to accommodate for such risks.

This thesis responds to these difficulties by attempting to realise informally, now, selective examples of the knowledge, practices and problems that may be realised formally, later. Specifically, it suggests to a reader a different but related objective - informal Engineering - and seeks to facilitate its comprehension. The conception papers suggest the formulation of knowledge that is general *and* formal. In contrast, this thesis takes the generality and formality of Engineering knowledge to be distinct and independent, and suggests the formulation of general knowledge, which pertains to *classes* of design Problem, but which remains informal. It develops informal, but general specifications in response to informal, but general requirements, and acquires and applies informal, but general substantive knowledge about informal, but general design Problems.

Having suggested to the reader the different, but related objective of informal Engineering, comprehension of this objective is supported in two ways. First, an example of carry forward in the manner sought is reported, and thus a reader is *shown* the 'technically more advanced' way of conducting HCI devised by the author (and which the reader is encouraged to better)[2]. Second, the difference between explicit Craft and informal Engineering is highlighted, both in the specification of alternative forms of HCI, and in case study reports. This difference concerns the address of classes of design Problem and instances of classes of design Problem in informal Engineering, rather than poorly related problem instances in explicit Craft. The reader may comprehend the objective of informal Engineering by contrasting the manner of carry forward sought with approximately equivalent (explicit Craft) knowledge and practices with which he or she is presumably already familiar.

---

[2]With respect to comprehension, a notable weakness of the example basic, application and design representations developed in this thesis is the selectiveness/incompleteness of these representations. Such selectivity results from the limited scope of the pre-conception, which is in turn required by the combination of inadequate knowledge of knowledge-practice-problem relations and limited resources (see section 1.2). The highly selective examples of carry forward reported in this thesis are not claimed to be the most instructive examples that could conceivably be devised. The examples provided are, currently, and for pragmatic reasons, the only examples that may be provided. Nevertheless, these examples are likely to enable a reader to acquire by ostension at least a preliminary view of the nature of informal Engineering.

With respect to the difficulty of lack of strategy, informal Engineering is envisaged as developing iteratively, that is, through repeated attempts to realise ever more effective combinations of knowledge, practices and problems, guided by assessments of the effectiveness, in practice, of the combination of discipline elements realised, and the refinement of the specification of this form of HCI (see Figure 1.11). This thesis does not attempt to specify completely or coherently an 'ideal' form of informal Engineering, or even the range of options for combinations of informal Engineering knowledge, practices and problems. It is for the reader to use the framework offered in section 1., together with other critiques of the state of the Art and the nature of the HCI enterprise etc., to devise additional types of basic, application and design representations, and to persuade the HCI community of the effectiveness, in practice, of carry forward involving such representations. Given the preliminary nature of the informal Engineering attempted in this thesis, the development of such 'more effective', 'technically more advanced' forms of informal Engineering are expected and encouraged.

With respect to the difficulty of excessive risk, this thesis suggests that the need for technical progress may be less pressing than suggested in the conception papers. Some technical progress, it is suggested, has been made - from implicit Craft to explicit Craft - and the effectiveness of the latter is suggested to be greater than that of the former. Further, although explicit Craft may be far from ideal, it may also be effective enough, at least for some purposes some of the time (witness the widespread development and application of Human Factors guidelines and user interface styles and the rapid spread of Information Technology). By characterising and pursuing informal (rather than formal) Engineering, the thesis also suggests that other objectives, which are less risky than those suggested in the background papers, could be pursued if preferred[3]. Achievement of these

[3]The development of informal Engineering is taken to be less risky, since it was assumed that HCI concerns need be less determinate for informal Engineering to succeed (see section 1.1.4), and since the innovation over the current state of the Art has been reduced. To advance from explicit Craft to formal Engineering, HCI must cease to address informal instances of design problems and come to address formal classes of design Problem. To advance from explicit Craft to informal Engineering, in contrast, HCI must only cease to address instances of design problems and come to address classes of design Problem, but these Problems may remain informally expressed.

objectives would constitute technical progress, albeit it of a less revolutionary kind. In the manner of carry forward proposed, knowledge about *classes* of design Problem may be applied to the development and instantiation of general (*class*) requirements and specifications, whereas, hitherto, only knowledge about *poorly related* problems has been applied to the development of *instance* requirements and specifications. The risk associated with the development of informal Engineering is also reduced by the clarification of the objective (by example), and by the suggestion of a research strategy (Zero-->build).

It seems reasonable to regard this thesis, then, as a considered, and constructive response to declarations of intent and the initial conceptual framework offered in the background conceptions. In this context, this thesis may be regarded as a contribution to: (i) a documented debate about the nature and effectiveness of HCI as a discipline, and the options for its development; and (ii) a program of research that seeks to realise and report technical progress.

Other possible future contributions to this debate and program of research are considered in the following section.

5.2. Possible Future Work

When considered as independent work, rather than a response to previous publications, the thesis may be seen to exhibit a number of limitations. Such limitations are an inevitable implication of the Zero-->build research strategy. Perception of such limitations is also encouraged, since it supports subsequent iterations in the progressive evolution towards informal Engineering.

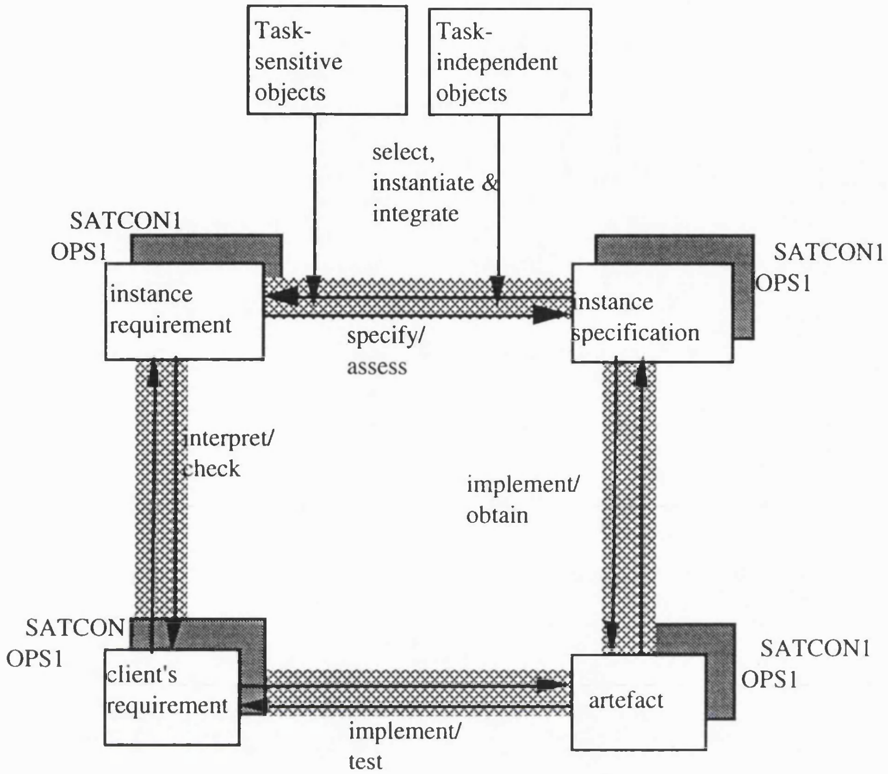
First, and with respect to the analysis of design Problems, the narrow-ness and high level of description of the pre-conception severely constrained its ability to support design. It was sometimes necessary to apply the pre-conception somewhat awkwardly. For example, a beach is not, intuitively, a kind of equipment, although it was conceived as such in chapter 4. (see section 4.1.3). It was also sometimes necessary to conduct additional analysis of the design Problem at hand in great haste and immediately prior to its application. For example, when analysing planning worksystems to support the design of the generic menu for planning tasks, a brief analysis of planning behaviours was conducted and instantiated for OPS1 and SATCON1 (see section 4.1.3). Future work may seek to remedy the limitations of the pre-conception by conducting further domain analysis following the Zero-->build strategy, that is, rapidly and easily, through literature review, and discussion with domain experts. Such analysis may, initially, concern peripheral, low-level characteristics of the domains of armed-conflict and of plans for armed-conflict (see section 1.2.1.). For example, the 'environment' may be considered as an additional domain object at the level of friends, enemies and neutrals. Alternatively, the nature of the 'systems' that comprise a neutral may be further analysed, or the 'effect' of a military plan may be studied (see sections 2.1.1.1 and 2.1.2.1). Particularly informative sources for such work may include texts and training material that concern military forces participating in 'non-combat' operations, such as peace-keeping, counter-insurgency and humanitarian interventions. Further domain analysis following the Zero-->build strategy may also seek to make explicit additional relations between concepts. Currently, the pre-conception expresses only the inseparability of planning and control, the reciprocity of power/threat and vulnerability, and the 'whole<-->part of' relations between domain objects and their components. More elaborate, complex relations could also be identified, such as relations between attributes at different levels of abstraction in the domain, or between the attributes of different objects within a level. Domain analysis may also be extended to other domains, both within C^2 and beyond. For example, section 2.1 envisaged the relevance of domains such as 'military picture compilation' and 'military communication'. Section 4.1.3 also raised the possibility of a domain of 'personal computer file control'. With respect to the analysis of domains beyond C^2, it was notable that the development and application of the generic menu structure for planning tasks suggested that a single artefact may contribute to the performance of many types of task. For example, on initial inspection, the demonstrator systems which supported the development of OPS1 and SATCON1 required users labelled 'planners' to perform, in addition to 'planning' tasks, 'domain modelling' tasks, 'database retrieval' tasks, 'personal computer file control' tasks and 'system configuration' tasks. Systems labelled 'planning-aids' may, on closer analysis, be found to be associated with many types of work, not just planning. The development of a specification of a class of artefact, then, may require the analysis of a number of design Problems, and not just the

Problem that, on initial inspection, appears to best characterise 'what the artefact does'. Finally, with respect to limitations of this thesis and worksystems, this thesis explicitly focussed on the analysis of domains. Future work may also consider C^2 worksystems more extensively, and seek to integrate worksystem and domain models.

With respect to limitations of this thesis and carry forward in informal Engineering evaluation, in the case reported, the costs incurred developing and applying the pre-conception were judged not to be justified by the value of evaluating an instance of a class of interaction. At best, the informal Engineering evaluation made explicit immediately, some aspects of effectiveness which the explicit Craft evaluation may not have made explicit until later, for example, when users had free access to a prototype. In the context of late, summative evaluation, then, a 'purpose for which the pre-conception is fit' remains to be found. Future work may seek to identify such a purpose, or more specifically, following the Zero-->build strategy, attempt to conduct evaluation in an informal Engineering manner, in which the opportunity to evaluate instances of (informal) classes of interaction may be of greater value. For example, such an opportunity may be of particular value in the context of the development of laboratory facilities dedicated to the evaluation of C^2 systems. Since the intention is to use such facilities to repeatedly evaluate any C^2 system with minimal re-configuration of the laboratory, the laboratory would presumably be designed to support the evaluation of instances of C^2 systems (the class). An informal Engineering evaluation, would enable *additional* reasoning about the completeness of the data collection and analysis tools to be provided within the laboratory - with reference to the super-ordinate characterisation of C^2 systems and their effectiveness in generally (see section 3.3.1.2). An explicit Craft evaluation, in contrast, would only generate a range of alternative data collection and analysis tools, and consider which of such tools would be utilised in the evaluation of randomly selected C^2 system instances. The opportunity to evaluate instances of (informal) classes of interaction may also be of value in the context of the development of generic GUI objects (see chapter 4). Such GUI objects are claimed to support a *class* of interaction, rather than a bespoke instance of interaction, and so developers of the object, object users (designers of instances), and end -users are concerned with the effectiveness of such objects for a *class* of interaction, and not just its effectiveness in poorly related instances. It is notable that in the case of informal Engineering evaluation reported, OPS1 is a bespoke artefact, which is poorly related to other artefacts, and it was not evaluated using the facilities and tools of a laboratory dedicated to C^2 systems. In a different design context, the assessment of alternative manners of carry forward in evaluation may have been different.

Third, with respect to limitations of this thesis and carry forward in informal Engineering specification, the selective nature of the case reports may be seen as ambiguating its conceptualisation. That is, the incompleteness of the general requirement and general specification developed in section 4.1.2. (incompleteness which was necessitated by the lack of relevant GUI objects at the time) blurs the distinction between general requirements and specifications, and generic GUI objects and their associated generic Problem elements. That is, the carry forward conducted in chapter 4 could have been alternatively, and reasonably, conceptualised as an example of explicit Craft, rather than informal Engineering. The highly selective general specification could alternatively have been conceived as just another type of GUI object - a type that is similar in nature and application to objects contained within conventional interface styles, but which is task-sensitive rather than task-independent. Given the limited number of generic objects in fact applied, it may not have been strictly necessary to have developed an explicit general specification to represent the product of selecting, specialising and integrating generic objects. It may have been more efficient to have simply applied the task-sensitive objects directly to the instances, in conjunction with the task-independent, interface-style objects and without and intermediate, and highly selective general specification of planning systems as a class (see Figure 5.1).

Figure 5.1: An Alternative Conceptualisation of Design Work Conducted in Chapter 4



Future work may seek to resolve this ambiguity by seeking to develop complete and coherent general requirements and general specifications for selected classes of artefact. Complete and coherent general specifications are required to maximise the value of the additional means of reasoning about completeness provided by informal Engineering - reasoning about completeness with respect to a super-ordinate class fails if the super-ordinate class is, itself incomplete. Complete specification would also distinguish 'general specifications' from 'generic objects', since the latter are defined as only partial (see section 4.1). The optimal degree of completeness of general requirements and specifications, however, remains to be established. Indeed, the experience of developing the general menu structure for planning systems (see section 4.1) suggests that a major issue for such work will be the nature of an appropriate trade-off between scope and generality, that is, for different classes of artefact, which features of an interaction are best specified at which level of generality. Given the the emergence of other generic GUI objects for planning systems, it seems reasonable to expect such work to achieve considerably more completeness than has been achieved in this thesis.

It is also notable that this thesis developed general knowledge that was 'prescriptive' with respect to instance specifications by providing informal, general, partial specifications (generic objects), rather than informal design principles. No argument against the development of informal principles is made - only an argument in favour of generic objects. Consequently, future work may seek to formulate informal design principles. Such work may commence by attempting to make explicit informal principles that supported the development of the general specifications for planning systems.

With respect to limitations of this thesis and the general objective of realising and reporting a more advanced manner of carry forward, it is notable that the thesis has not offered a complete and coherent specification of informal HCI Engineering. As indicated in section 5.1, this thesis seeks only to highlight the differences between alternative forms of HCI in order to characterise the progress which has already been made, which is sought here, and which may be made in the future. The development of such a complete specification may be necessary to offer the HCI community, and the other communities to which it relates, unambiguous accounts of the state of the HCI Art and the outputs of research. However, the immediate development of such a specification is not proposed here, since it would be incompatible with the Zero-build strategy. Zero-->build calls for the gradual refinement of this specification to reflect, and guide gradual, and incremental progress, rather than the adoption of 'specifying the discipline of HCI' as an end in itself.

5.3. Possible Criticisms of this Thesis

In addition to the limitations suggested in the previous section, a number of possible criticisms of this thesis may also be made. A brief consideration of these criticisms, and their counter arguments, may be instructive, and may also serve to bring this thesis to a close.

First, it could be argued that there is little difference between informal Engineering and poor Engineering. Informal Engineering is just formal Engineering executed badly. However, this criticism mistakes the initial, selective examples of informal Engineering presented in this thesis, with the more complete and coherent examples of informal Engineering, that may be achieved later as a result of iteration and progressive evolution. In this thesis, the basic, application and design representations developed are selective, because the Zero-->build strategy (which requires immediate consideration of a wide range of issues) was adopted, not because informal Engineering knowledge is necessarily selective. Selective examples are just the only kind of examples that may be achieved with the limited resources of a PhD. Should more resources become available, more complete and coherent examples of informal Engineering knowledge could be developed. Indeed, such development is envisaged (see Figure 1.12, Figure 1.15, and section 5.2). The criticism that informal Engineering is poor Engineering also fails to recognise that comprehension of knowledge-practice-problem relations is unlikely to be *permanently* inadequate. The Zero-->build strategy was adopted for this thesis because current comprehension of knowledge-practice-problem relations was considered to be inadequate. Consequently, this thesis intentionally prioritised the development of basic representation's that are 'fitness-for-purpose' (supports efficacious and efficient design practice) over the development of basic representations that are explicit, coherent, simple, accurate and well-scoped. However, once effective combinations of knowledge, practices and problems are established, these priorities may change. For example, if 'fitness-for-purpose' may be presumed, then simplicity, accuracy and scope may become more important, and a more 'top-down' research strategy may be adopted - a strategy that would be expected to produce the 'high-quality' informal Engineering knowledge ultimately desired.

Finally, it could also be argued that this thesis has mis-represented the current state of the Art. In fact, current best practice already addresses informal classes of design Problem and informal instances of classes of Problem, and simply labelling such work 'informal Engineering' is of no assistance. However, it is difficult to support such claims with examples of HCI work that clearly exhibit the characteristics of informal Engineering. Some work may appear to constitute informal Engineering, because it is characterised using similar terms to those used here. However, similar terms are in fact used in different senses. For example, Shackel (1986) uses the term 'general' to mean 'frequent or widespread', 'common or shared', 'high-level', 'imprecise' or 're-usable', and not 'class'. The range of problems to which Shackel's 'general' view of effectiveness applies, the nature of these problems, and relations between these problems, is specified only poorly[4]. Consequently, the claim that, for Shackel, 'general' means 'applicable to a class of Problem' is difficult to support. In this thesis, in contrast, the term 'general' is used only to express 'class' relationships. These relationships are made explicit in a Problem hierarchy (see Figure 3.4) and the Problems related in the hierarchy are analysed to a useful extent (see Figure 2.1). As another example, some work uses the term 'general' to refer to re-use *within* an artefact, rather than re-use *across* artefacts. For example, McCracken & Akscyn (1985) use configurations of bespoke buttons and fields to construct a number of screen displays for a single database system 'ZOG'. In this thesis, in contrast, a general specification was used to specify two distinct planning systems - OPS1 and SATCON1. The discipline status of other work is often highly ambiguous, because the work is typically reported for reasons other than characterising its discipline status. For example, Denley et al. (submitted for publication) report an evaluation of a menu structure for dynamic telecommunications service control tasks which is instantiated for two services (surveillance and tele-conferencing) and which, on initial inspection, resembles a task-sensitive generic object (see section 4.1.1). However, the general problem element with which this object is associated is not reported, so the nature and extent of the object's generality is not fully explicit. Nor is it suggested whether the object evaluated is intended to support the development of a general specification, or an instance specification, or both[5]. Thus, Denley et al.'s work may be construed as an example of explicit Craft, informal Engineering, explicit Craft supporting informal Engineering, or informal Engineering supporting explicit Craft, or all four. Given that key terms may have multiple meanings, and that previous work is typically reported for reasons other than establishing its discipline status, blunt assertions about the state of the Art should not be accepted. Supporting evidence is required.

With respect to assistance for a reader, this thesis is a resource for intelligent, situated HCI action, so it is difficult to delimit the uses that a reader may construct for objectives expressed at a discipline level. However, it is possible to suggest that consideration of objectives at this level may assist attempts to achieve 'progress', rather than 'growth', that is,

---

[4]These problems and their relations are nevertheless stated with a degree of precision that is adequate for many purposes.
[5]The menu structure and its evaluation deserve interest nevertheless.

attempts to realise innovative forms of HCI, rather than conduct conventional forms of HCI in relation to innovative kinds of Information Technology or work domain. Consideration of objectives at the discipline level, then, may help to distinguish these different objectives as distinct options. Consideration of objectives at a discipline level may also help to define the conditions under which the Zero-->build strategy may be applicable. This strategy was devised to achieve discipline progress and, as indicated earlier this section, other strategies may be more appropriate when seeking growth. Presumably, the knowledge-practice-problem relations applicable to conventional forms of HCI are already adequately comprehended. Consideration of objectives at the discipline level, then, may also help to justify the use of Zero-->build and guide its appropriate application. In addition, consideration of objectives at the discipline level may help to identify means of carry forward that are likely to be relatively easy to achieve (explicit Craft means), from alternatives that are likely to be more difficult to achieve, even in the medium term (informal Engineering means). Finally, however, the most pertinent assistance that consideration of objectives at the discipline level may offer those involved in military procurement, is to emphasise that attempts to improve a manner of carry forward consistitute a very considerable undertaking. Such attempts strike at the very heart of a discipline, and are not to be taken lightly.

References

AHMAD, K., HOLMES-HIGGINS, P. R., HORNSBY, C. P. W. and LANGDON, A. J. 1988, Expert Systems for Planning and Controlling Physical Networks in the Water Industry, *Knowledge-Based Systems*, **1**(3), pp 153-165.

ALBEN, L., FARIS, J. & SADDLER, H. 1994, Making it Macintosh: Designing the Message When the Message is Design, *Interactions*, Jan 94, pp11-20.

ALGER, J. I. 1982, *The Quest for Victory: The History of the Principles of War*, (London: Greenwood Press).

APPLE COMPUTER INCORPORATED, 1985, Inside Macintosh Vol I. (Wokingham, England: Addision-Wesley).

APPLEGATE, C., ELSAESSER, C. and SANBORN, J. 1990, An Architecture for Adversarial Planning, *IEEE Trans. on Systems, Man and Cybernetics*, **20**(1), pp 186-194.

BAECKER, R. M. & BUXTON, W. A. S. 1987, 'The Star, the Lisa and the Macintosh'. In Baecker, R. M. & Buxton, W. A. S. (eds.) *Readings in Human-Computer Interaction: A Multidisciplinary Approach*, (Los Altos, CA, USA:Morgan Kaufmann) pp 649-652.

BARNARD, P. J. & GRUDIN, J. 1988, 'Command Names'. In Helander, M. (ed.) *Handbook of Human-Computer Interaction*, (Amsterdam:North-Holland) pp237-255.

BARNARD, P. J., HAMMOND, N. V., MORTON, J. & LONG, J. B. 1981, Consistency and Compatibility in Human-Computer Dialogue, *Int. J. Man-Machine Studies*, 15, 87-134.

BARNARD, P., WILSON, M. & MACLEAN, A. 1988, Approximate Modelling of Cognitive Activity with an Expert System: A theory-Based Strategy for Developing an Interactive Design Tool *The Computer Journal*, 31(5), pp445-456.

BELLOTTI, V. 1988, Implications of Current Design Practice for the Use of HCI Techniques. In D. M. Jones & R. Winder (eds) *People and Computers IV*, (Cambridge: Cambridge University Press), 13-34.

BELLOTTI, V. 1993, Integrating Theoreticians' and Practitioners' Perspectives with Design Rationale, Proceedings of INTERCHI'93: Human Factors in Computing Systems (ACM Press), pp 101-106.

BEWLEY, W. L., ROBERTS, T. L., SHROIT, D. & VERPLANK, W. L. 1983, 'Human Factors Testing in the Design of Xerox's 8010 "Star" Office Workstation. In (eds.) *Proceedings of CHI'83*, .

BOEHM, B. W., BROWN, J. R., KASPAR, H., LIPOW, M., MACLOED, G. & MERRIT, M. 1978, *Characteristics of Software Quality*, TRW Series of Software Technology (Amsterdam: North Holland).

BORNAT, R. & THIMBLEBY, H. 1989, 'The Life and Times of Ded, Text Display Editor', in Long & Whitefield (eds.) *Cognitive Ergonomics and Human-Computer Interaction* (Cambridge, UK: Cambridge University Press).

BRAIM, S. P. & HEPWORTH, N. 1992, 'A Fully-Featured Computer Interface for Intelligence and Command & Control Applications'. In *IDASCO, '92*, Proceedings of the IEE's Conference on Information Decision Action Systems in Complex Organisations, Oxford, England, 6-8 April 1992, Conference Publication No. 353, (London: Institute of Electrical Engineers) pp 149-152.

BROWN, C. M. 1989, Human-Computer Interface Design Guidelines (Norwood, NJ, USA: Ablex).

BROWN, G. G., GOODMAN, C. E. & WOOD, R. K., 1990, Annual Scheduling of Atlantic Fleet naval Combatants, *Operational Research*, 38(2), pp249-259.

BUCKINGHAM SHUM, S. & HAMMOND, N. 1994, Delivering HCI Modelling to Designers: a Framework and Case Study of Cognitive Modelling, *Interacting with Computers*, **6**(3), pp 311-341.

BUCKLEY, P. 1994, The GUI is Too Thin for Domain Support, IEE Colloquium on "What are Graphical User Interfaces Good For" 27th January, 1994 Savoy Place, London.

BURNS, T. 1969, Models, Images and Myths. In Marquis, E. G. & Gruber, W. H. (eds.) *Factors in the Transfer of Technology* (Cambridge, Mass.: M.I.T. Press).

BURTON, J. W. 1968, *Systems, States, Diplomacy and Rules*, (Cambridge, UK: Cambridge University Press).

BUXTON, B. 1993, 'HCI and the Inadequacies of Direct Manipulation Systems'. *SIGCHI Bulletin*, 25(1) pp21-22.

CARROLL, J. 1985, *What's in a Name?*, (New York: Freeman)

CARROLL, J. M. & ROSSON, M. B. 1985, Usability Specifications as a Tool in Iterative Development. In H. R. Hartson, (Ed.) *Advances in Human-Computer Interaction*, 1, (Norwood, NJ: Ablex) pp 1-28.

CARROLL, J. M., MACK, R. L. & KELLOGG, W, A. 1988, Interface Metaphors and User Interface Design. In Helander, M. (ed.) *Handbook of Human-Computer Interaction*, (Amsterdam:North-Holland) pp67-85.

CHANDRESEKARAN, B. JOSEPHSON, J., KEUNEKE, A. & HERMAN, D. 1989, 'Building Routine Planning Systems and Explaining their Behaviour'. *International Journal of Man-Machine Studies*, 30, pp377-398.

CHECKLAND, P. (1981) *Systems Thinking, Systems Practice.* (Chichester, U.K.:Wiley).

CLANCEY, T. 1987, *Red Storm Rising*, (Glasgow, UK:Fontana/Collins).

CLAUDE, Inis L. 1988, *States and the Global System: politics, law and organisation*, (Basingstoke, UK: MacMillan).

CLAUSEWITZ, K. VON, 1832, *On War*, (Harmondsworth, Mddsx. England: Penguin) reprinted in 1982.

COLBERT, M. 1993, Report on Validation of IRT Experiments, Ergonomics Unit Internal Report, ACT/93/Deliverable 21, University College London, 26 Bedford Way, WC1H 0AP.

COLBERT, M., DOWELL, J., LONG, J. B. & McCUBBIN, J. B. 1992, Alternative Views of the Quality of Military Plans. In *IDASCO, '92*, Proceedings of the IEE's Conference on Information Decision Action Systems in Complex Organisations, Oxford, England, 6-8 April 1992, Conference Publication No. 353, (London: Institute of Electrical Engineers) pp177-181.

COLLINS, H. M. 1982, Tacit Knowledge and Scientific Networks. In BARNES, B. & EDGE, D. (eds.) *Science in Context* (Milton Keynes:Open University Press) pp44-64.

CONLEY, R. E. 1982, Military Command and Control (C^2), in J. Hwang, D. Schutzer & K. Shere (eds) *Selected Concepts in Command & Control*, (London: Gordon & Breach), pp23-32.

CORBETT, J. S. 1911, *Some Principles of Maritime Strategy*, (New York:AMS) reprinted in 1972.

CRAIG, J. C. 1994, Database Developer's Guide with Visual-Basic3 (Redmond, Wash., USA: Microsoft).

CURRIE, K. & TATE, A. 1991, 'O-Plan: the open planning architecture'. *Artificial Intelligence*, 52, pp49-86.

de SOUZA, F. L., LONG, J. B. & BEVAN, N. 1990, Types of Error and Difficulty in Using Human Factors Guidelines: The Case of Interface Menu Design. In Lovesy, E. J. (ed.) *Contemporary Ergonomics 1990*, Proceedings of the Ergonomics Society's Annual Conference, Leeds, England, 3-6 April, 1990 (London: Taylor Francis). pp 340-346.

DENLEY, I. & LONG, J. 1990, A Framework for Evaluation Practice, in E. J. Lovesey (ed) *Contemporary Ergonomics 1990*, Proceedings of the Ergonomics Society's Annual Conference, Leeds, England, 3-6 April 1990, (London:Taylor Francis), pp31-37.

DENLEY, I., HEDMAN, L. R., HILL, B., WHITEFIELD, A., PAPADOPOULOUS, K, CLARKE, A & HINE, N. 1993, Usability Principles for Service Design. In Byerley, P. F., Barnard, P. J. & May, J. (eds.) *Computers, Communication and Usability; Design Issues, Research and Methods for Integrated Services*, North-Holland Studies in Telecommunication Volume 19 (Elsevier North-Holland:Amsterdam) pp291-342.

DENLEY, I., HILL, B. & WHITEFIELD, A. (submitted for publication) An Experimental Evaluation of a Normative Interface Design for the Configuration of Telecommunication Services. Ergonomics & HCI Unit, University COllege London, 26 Bedford Way, London WC1H 0AP.

DENLEY, I., WHITEFIELD, A., BYERLEY, P., VOIGT, U-B., HERMANN, S. & MAY, J. 1992, 'Design Principles for Improving Service Integration for End-Users in Broadband Communication Systems'. In Monk, A., Diaper, D. & Harrison, M. D. (eds.) People and Computers VII, Proceedings of the HCI'92 Conference, York, September, 1992, (Cambridge:Cambridge University Press), 307-322.

DILWORTH, C. 1981, *Scientific Progress*, (Dordrecht: Reidel).

DOWELL, J. 1993, Cognitive Engineering and A Rationalisation of the Flight Strip PhD. Thesis University College London, Gower St., London, U.K.

DOWELL, J. & LONG, J. B. 1989, Towards a Conception for an Engineering Discipline of Human Factors, *Ergonomics* , **32**(11), 1513-1535.

DOWLE, S. W. 1989, Domain Modelling for Requirements Specification, in *Proceedings of 3rd IEE Conference on Command, Control, Communication and Management Information Systems*, Bournemouth, 2nd-4th May, 1989, Conference Publication No. 304. (London:Institute of Electrical Engineers), 1-7.

EASON, K. D. 1984, Towards the Experimental Study of Usability, *Behaviour & Informaion Technology*, **3**(2), 133-145.

EVANS, M. H. H., 1990, *Amphibious Operations*, (London:Brassey's).

FOWLER, C., MACAULAY, L., CASTELL, A. & HUTT, A. 1989, An Evaluation of the Usability of a Human Factors based Requirements Capture Methodology, in Sutcliffe, A. & Macaulay, L. (eds.) *People and Computers V*, Proceedings of the Fifth Conference of the British Computer Society Human-Computer Interaction Specialist Group, Nottingham, 5th-8th September, 1989, (Cambridge:Cambridge University Press).

GALE, S. & BRENNAN, C. 1993 Opening Windows with Corporate Style. Delegates handout at a seminar organised the British Computer Society Human-Computer Interaction Specialist Group, London, 28th May, 1993. BT Research Laboratories, MLB Room 2/10, Martlesham Heath, Ipswich, IP5 7RE.

GARDINER, M. M. & CHRISTIE, B. 1987, *Applying Cognitive Psychology to User Interface Design*, (John Wiley & Sons:Chichester, UK).

GAYE, C. & McCUBBIN, J. 1991, The Application of Knowledge Based Systems to Amphibious Command & Control, in *Proceedings of Defence Oceanography*.

GOULD, J. D. 1987, How to Design Usable Systems, in H-J. Bullinger & B. Shackel (eds.) *INTERACT, '87*, (Elsevier: North-Holland), 35-39.

GOULD, J. D. & LEWIS, C. 1983, Designing for Usability - Key Principles and What Designers Think, in *CHI'83* Proceedings, ACM, pp50-53.

GREEN, T. R. G., 1989, Describing Information Artefacts with Cognitive Dimensions and Structure Maps. In D. Diaper & N. Hammond (eds), *People and Computers VI*, Proceedings of the Sixth Conference of the BCS-HCI Specialist Group, 20-23 August, 1991 (Cambridge, U.K.:Cambridge University Press), 297-316.

GRIEF, I. 1985, Microcomputer User Interface Toolkits: the Commercial State of the Art. In Borman. L. & Curtis, B. (eds.) *Human Factors in Computing Systems -II*, Proceddings of the ACM/SIGCHI CHI'85 Conference, San Francisco, CA, USA, (Amsterdam:North Holland) pp225.

GROVES, E. 1990, *The Future of Sea Power*, (London: Routledge).

GRUNDY, A. J. 1988, Humans, Computers and Contracts, in D. M. Jones & R. Winder (eds) *People and Computers IV*, (Cambridge: Cambridge University Press), 161-175.

HAKIEL, S. 1993, What are GUIs Good For? Call for papers, IEE Computing and Control Division, Colloquium London, 27th February, 1994. IBM UK Laboratories, Hursley Park, Winchester, Hampshire SO1 2JN.

HARRIS, C. J. & WHITE, I. 1987, (eds) *Advances in Command, Control and Communications Systems*, (London: Peregrinus).

HILL, B. H., LONG, J., SMITH, W. & WHITEFIELD, A. 1993, Planning for Multiple Task Work - Analysis of a Medical Reception Worksystem. In Proceedings of INTERCHI'93, Amsterdam, 25-29 April 1993. (Reading, MA: Addison-Wesley.)

HOC, J. M. 1988, *Cognitive Psychology of Planning*, (London: Academic Harcourt Brace Jovanovich).

HOLMAN, R., YOUNG, C. & DORRINGTON, J. 1987, The Specification of Surface Naval Command Systems, in *Proceedings of 2nd IEE Conference on Command, Control, Communication and Management of Information Systems*, Bournemouth, 1st-3rd April, 1987, Conference Publication No. 275. (London:Institute of Electrical Engineers), 16-25.

HORTON, W. K. 1990, Designing and Writing On-Line Documentation (New York: Wiley).

HOWARD, S. & MURRAY, D. 1987, A Taxonomy of Evaluation Techniques for HCI, in Bullinger, H. J. & Shackel, B. (eds.) *INTERACT'87* Proceedings of the second IFIP Conference on Human-Computer Interaction, Stuttgart, Germany, September 1987 (Amsterdam:North-Holland) pp453-460.

HUTCHINS, E. L., HOLLAN, J. D. & NORMAN, D. A. 1985, Direct Manipulation Interfaces, *Human-Computer Interaction*, 1(4), pp311-338.

HUTT, A., DONNELLY, N., MACAULAY, L. A., FOWLER, C. J. H. & TWIGGER, D. 1987, Describing a Product Opportunity: A Method of Understanding the Users' Environment, in D. Diaper & R. Winder (eds) *People & Computers III*, (Cambridge, UK: Cambridge University Press).

HWANG, J., SCHUTZER, D. & SHERE, K. (eds) 1982, *Selected Concepts in Command & Control*, (London: Gordon & Breach).

IBM, 1992, Object-Oriented Interface Design: Common User Access

JACOB, R. J. K., LEGGETT, J. J., MYERS, B. A. & PAUSCH, R. 1993, Interaction Styles and Input/Output Devices, Behaviour & Information Technology, 12(2), pp69-79.

JOINT CHIEFS OF STAFF 1988, *The Official Dictionary of Military Terms*, (Philadelphia: Hemisphere).

JONES, J. C. 1970, *Design Methods* (London: Wiley).

JORDAN, G., LEE, I. & CAWSEY, G. 1988, *Learning from Experience*, Report to the Minister for Defence Procurement, (London: HMSO).

JORDAN, P. W., DRAPER, S. W., MACFARLANE, K. K., & McNULTY, S-A. 1991, Guessability, Learnability and Experienced User Performance. In D. Diaper & N. Hammond (eds), *People and Computers VI*, Proceedings of the Sixth Conference of the BCS-HCI Specialist Group, 20-23 August, 1991 (Cambridge, U.K.:Cambridge University Press), 237-245.

KANNEGAARD, J., HOEBER, T. & PROBST, R. 1988, 'Special Section: OPEN LOOK User Interface', *Sun Technology*, 1(4), pp 57-86.

KELLOG, W. A. 1987, Conceptual Consistency in the User Interface: Effects on User Performance. In: H. J. Bullinger and B. Shackel (Eds.) *Human Computer Interaction - INTERACT '87.* (North-Holland: Elsevier Science Publishers B.V.)

KIELY, D. G., 1988, *Naval Surface Weapons*, (London:Brassey's).

KLEIN, G. A. & BRESOVIC, C. P. 1986, Design Engineers and the Design Process: Decision Strategies and the Human Factors Literature. In Proceedings of the Human Factors Society 30th Annual Meeting.

LANIR, Z. 1991, The Reasonable Choice of Disaster, In Rasmussen, J., Brehmer, B. & Leplat, J. (eds.) *Distributed Decision Making:Cognitive Models for Cooperative Work* (Chichester, U.K.:John Wiley & Sons) pp217-230.

LAWRENCE, T. E. 1935, *Seven Pillars of Wisdom*, (Penguin: London).

LIFE, M.A., LEE, B.P. and LONG, J.B. 1990. Assessing the Usability of Speech Interfaces for Future Battlefield Computers: A Method and an Illustration of its Use. In Proc. Voice Systems Worldwide '90, London, 30 May-1 June 1990. (New York: Media Dimensions) 121-125.

LOBERG, G., POWELL, G. M., OREFICE, A. and ROBERTS, J. D. 1986, Representing Operational Planning Knowledge, *IEEE Trans. on Systems, Man and Cybernetics*, **16**(6), pp 774-787.

LONG, J. B. 1987, Cognitive Ergonomics and Human-Computer Interaction. In Warr., P. (ed.) Psychology at Work, (Penguin: England).

LONG, J. & DOWELL, J. 1989, Alternative Conceptions of the Discipline of Human-Computer Interaction, in M. Harrison & A. Monk (eds) *People and Computers: Designing for Usability*, Proceedings of the Second Conference of the BCS-HCI Specialist Group, York, September 23rd-26th, 1989, (Cambridge, UK: Cambridge University Press), 23-26.

LIM, K. Y., LONG, J. B. & SILCOCK, N. 1992, Integrating Human Factors with the Jackson Systems Development Method: an Illustrated Overview, *Ergonomics*, 35(10) pp1135-1161.

MACLEAN, A., YOUNG, R. M., BELLOTTI, V. & MORAN, T. 1991, Questions, Options and Criteria: Elements of a Design Space Analysis' *Human-Computer Interaction*, **6**(3/4), pp 201-250.

MAHONEY, K. & GOWER, A. 1991, The Development of a Visual Style for a BT X Window System Toolkit. In D. Diaper & N. Hammond (eds), *People and Computers VI*, Proceedings of the Sixth Conference of the BCS-HCI Specialist Group, 20-23 August, 1991 (Cambridge, U.K.:Cambridge University Press), 265-279.

McCRACKEN, D. L. & AKSCYN, R. M. 1984, Experiences with the ZOG human-computer interface system, *Int. J. Man-Machine Systems*, **21**(4), pp 293-310.

MICROSOFT CORPORATION, 1992, The Windows Interface: An Application Design Guide. (Redmond, WA: Microsft Press).

MILLET, A.R. & WILLIAMSON, M. 1988, *Military Effectiveness Vol I: the First World War*, (London: Allen & Unwin).

MOLL, K. L. 1982, The C^3 Functions, in J. Hwang, D. Schutzer & K. Shere (eds) *Selected Concepts in Command & Control*, (London: Gordon & Breach), 23-32.

MULVEHILL, A. M. 1988, A User Interface for A Knowledge-Based Planning and Scheduling System, in *IEE Transactions on Systems, Man and Cybernetics*, **18**(4), 514-521.

MUMFORD, E. 1979, *Designing Participatively.* (Manchester, U.K.: Manchester Business School).

NIELSEN, J. 1989, Usability Engineering at a Discount. in Salvendy, G. & Smith, M. J. (eds.) *Designing and Using Human-Computer Interfaces and Knowledge Based Systems* (Hillsdale, N.J., U.S.A.: Lawrence Erlbaum).

NOEL, J. V. & BEACH, E. L. 1988, *Naval Terms Dictionary*, (Annapolis Maryland: Naval Institute Press).

NORMAN, D. A. & DRAPER, S. W. 1986, *User-Centred System Design*, (London:Lawrence Erlbaum).

OHLFS, K. 1991, Designing and Icon. *Next World*, 4, 74-75.

OPEN SOFTWARE FOUNDATION, 1991, *OSF/Motif Style Guide*, (New York:Prentice Hall).

PAAP, K. R. & ROSKE-HOFSTRAND, R. J. 1988, 'Design of Menus'. In Helander, M. (ed.) *Handbook of Human-Computer Interaction*, (Amsterdam:North-Holland) pp205-235.

PEJTERSEN, A. M. 1991, Icons for Representation of Domain Knowledge in Interfaces. *Advances in Knowledge Organisation*, **2**, pp. 175-193.

PIETRAS, C. M., 1994, The Development of Cognitive Models of Planning for Use in the Design of Project Management Systems, *Int. J. Human-Computer Studies*, **40**, pp 5-30.

PINEDO, M., SHAW, D. & PING-CHANG YEN, B. 1993, Interactive Scheduling: Algorithms and User Interfaces, Paper presented at a National Science Foundation Workshop on Scheduling, January 1993.

RASMUSSEN, J. & VICENTE, K. J. 1989, Coping with Human Errors Through System Design: Implications for Ecological Interface Design, *International Journal of Man-Machine Studies*, **31**, 517-534.

RASMUSSEN, J. & VICENTE, K. J. 1990, Ecological Interfaces: A Technological Imperitive in High Technology Systems? *Int. J. HCI*, 2(2), 93-111.

REASON, J. 1991, Two Contrasting Military Plans: The Schlieffen Plan and Plan 17. In Rasmussen, J., Brehmer, B. & Leplat, J. (eds.) *Distributed Decision Making. :Cognitive Models for Cooperative Work* (Chichester, U.K.:John Wiley & Sons) pp205-214.

RICE, M. A. & SAMMES, A. J. 1989, *Communications and Information Systems for Battlefield Command & Control*, (London: Brassey).

ROSENBERG, J. K. & MORAN, T. P. 1984, 'Generic Commands'. In *INTERACT'84*, Proceedings of the First IFIP Conference on Human Computer Interaction, London, September, 1984, (Amsterdam:Elsevier), pp 360-364.

SALTER, I. 1994, The Design of Formal Languages, PhD. Thesis, University College London, Gower St., WC1H 0AP.

SANDERSON, P. M., 1989, The Human Planning and Scheduling Role in Advanced Manufacturing Systems: an Emergin Human Factors Domain. *Human Factors*, 31(6), pp 635-666.

SCAPIN, D. L. 1990 Organising Human Factors Knowledge for the Evaluation and Design of Interfaces. *Int. J. of HCI*, 2(3) 203-229.

SCHMUCKER, K. J. 1987, MacApp: An Application Framework. In Baecker, R. M. & Buxton, W. A. S. (eds.) *Readings in Human-Computer Interaction: A Multidisciplinary Approach*, (Los Altos, CA, USA:Morgan Kaufmann) pp591-594.

SHACKEL, B. 1986, Ergonomics in Design for Usability, in M. Harrison & A. Monk (eds) *People and Computers: Designing for Usability*, Proceedings of the Second Conference of the BCS-HCI Specialist Group, York, September 23rd-26th, 1986, (Cambridge, UK: Cambridge University Press), 44-64.

SHERWOOD-JONES, B., HOLT, J., FERGUSSON, N., NARBOROUGH-HALL, C. S. & BRENNAN, S. D. (1992), A Theory of C^2 Organisations, in Proceedings of Information Decision Action Systems in Complex Organisations, Oxford, UK, 6th-8th April, 1992, (London: Institute of Electrical Engineers).

SHNEIDERMAN, B. 1982, The Future of Interactive Systems, *Behaviour and Information Technology*, **1**(3), 237-256.

SIMON, H. A. 1981, *The Sciences of the Artificial*, Second Edition, (Cambridge, Massachusettes: Massachusettes Instutute of Technology Press).

SINGLETON, W. T. 1987, The Design of Design Teams in High-Technology Industries, *Applied Ergonomics*, 18(2), 111-114.

SLAGLE, J. R. & HAMBURGER, H. 1985 An Expert System for a Resource Allocation Problem, *Communications of the ACM*, **28**(4) 994-1004.

SMITH, S. L. 1986, Standards versus Guidelines for Designing User Interfaces Software. *Behaviour & Information Technology*, 5, 47-61.

SMITH, D. CANFIELD, IRBY, C., KIMBALL, R. VERPLANK, W. L. & HARSLEM, E. 1982, 'Designing the Star User Interface'. *Byte*, 7(4), 242-282.

SMITH, S. L. & MOSIER, J. N. 1984, The User Interface to Computer-Based Information Systems: a Survey of Current Software Design Practice. In *INTERACT'84*, Proceedings of the First IFIP conference on Human Computer Interaction, London, September 1984 (Amsterdam: North Holland) pp 35-39.

SMITH, S. L. & MOSIER, J. N. 1986, *Guidelines for Designing User Interface Software*, MITRE Corporation Report, ESD-TR-86-278.

SNYDER, K. M., HAPP, A. J., MALCUS, L, PAAP, K. R. & Lewis, J. R. 1985, Using Cognitive Models to Create Menus. In Proceedings of the Human Factors Society 29th Annual Meeting pp. 655-657.

STORRS, G. 1989, Towards a Theory of HCI: A Conceptual Model of Human-Computer interaction?, *Behaviour & Information Technology*, **8**(5), 323-334.

SUCHMAN, L. A. 1987, *Plans and Situated Actions*, (Cambridge: Cambridge University Press).

SUTCLIFFE, A. 1993 Software Re-Use: State of the Art and Survey of Technical Approaches. In Walton, P. & Maiden, N. (eds.) *Integrated Software Reuse: Management and Techniques*, (Aldershot, U.K: Ashgate), pp 51-76.

TAINSH, M. A. 1982, On man-computer dialogues with Alpha-numeric status displays for naval command systems, *Ergonomics*, **25**, 683-703.

TAINSH, M. A. 1985, Job Process Charts and Man-Computer Interaction withiin naval Command Systems, *Ergonomics*, **28**(3), 555-565.

TAINSH, M. A. 1992, On the Assessment and Evaluation of Designs for Human Computer Interaction. In E. J. Lovesey (ed) *Contemporary Ergonomics 1992*, Proceedings of the Ergonomics Society's Annual Conference, Birmingham, England, 7-10 April 1992, (London:Taylor Francis), pp346-351.

TETZCHNER, J. S. von 1993, Integration and Interface Design of Telecommunications Applications and Functions in a Multi-Service Telecommunication Environment, Degree Thesis, Dept. of Computer Science, University of Oslo.

TETZLAFF, L. & SCHWATRZ, D. R. 1991, The Use of Guidelines in Interface Design. In Robertson, S. P., Olson, G. M. & Olson, J. S. (eds.) *CHI'91 - Reaching Through Technology*, Proceedings of the Conference on Human Factors in Computer Systems, New Orleans, Louisiana, April 27-May 2nd, 1991 (Addison-Wesley: Reading, Massachusettes), pp 329-333.

THOMPSON, J. 1986, No Picnic (Isle of Man: Fontana/Collins).

TILL, G. 1987, *Modern Sea Power*, (London:Brassey's).

US DoD 1978, Report of the Defense Science Board Task Force on Command & Control Systems Management, July 1978, Office of the Under Secretary of Defense, Research & Engineering, Wahington.

US DoD 1988, Formal Investigation in to the Circumstances Surrounding the Downing of Iran Air FLIGHT 655 on 3rd July 1988. U.S. Dept. of Defense Investigation Report, Office of the Under Secretary of Defense, Public Affairs, Wahington.

VAN CREFELD, M. 1985, Command in War, (Cambridge Massachusettes: Harvard University Press).

VAN DER SCHAAF, T. W. 1993, Developing and Using Cognitive Task Typologies. *Ergonomics*, 36(11) pp. 1439-1444.

VAN PUTTEN, J. SCHARENBORG, N. & WOERLEE, A. 1993 A Generic user Interface Constructor for Planning and Scheduling Applications in ??? (eds) *People & ComputersVIII*, Proceedings of the Seventh Conference of the BCS-HCI Specialist Group, Loughborough, September 23rd-26th, 1993, (Cambridge, UK: Cambridge University Press), 143-156.

VICENTE, K. J. 1990, Coherence- and Correspondence-driven work domains: implications for system design. *Behaviour and Information Technology,* 9 (6), pp. 493-502.

WALTON, P. & MAIDEN, N. 1993, *Integrated Software Reuse: Management and Techniques* (Aldershot, U.K.:Ashgate)

WESTON, J. D. & HASLAM, R. A. 1992, How Successfully can Non-Ergonomists Recognise and Apply Ergonomic Principles?. In Lovesy, E. J. (ed.) *Contemporary Ergonomics 1992*, Proeedings of the Ergonomics Society's Annual Conference, Birmingham, England, 7-10 April, 1990 (London: Taylor Francis). pp 334-339.

WHITESIDE, J., BENNETT, J. & HOLZBLATT, K., 1988, Usability Engineering: Our Experience and Evolution. In Helander, M. (ed.) *Handbook of Human-Computer Interaction*, (Amsterdam:North-Holland) pp791-817.

WINCH, P. 1958, *The Idea of Social Science* (London:Routledge & Kegan Paul).

WIXON, D. 1992, A Review of Mayhew, D. J. Priniciples and Guidelines in Software User Interface Design (New York: Prentice Hall) in *SIGCHI Bulletin*, **25**(2) pp 51-53.

WOHL, J. G. 1981, Force Management Decision Requirements for Air Force Tactical Command and Control, *IEEE Transactions*, SMC-**11**(9).

WOODS, D. D. and ROTH, E. M. 1988, Cognitive Systems Engineering in M. Helander (ed.) *Handbook of Human-Computer Interaction*, Elsevier, North-Holland, pp 1- 43.