A Cognitive Analysis
of
Design Rationale Representation

Simon J. Shum

Submitted for the Degree
of
Doctor of Philosophy

University of York
Department of Psychology

December, 1991

Abstract

An emerging field of human-computer interaction research is addressing the problem of representing, as explicit *design rationale* (DR), the reasoning underlying the design of computer systems, user-centred rationale being of particular interest. Ideally, formalisms for DR should allow reasoning to be expressed in a form which is accessible to both humans and computers. Semi-formal notations which reify the structure of arguments, and can be manipulated as hypertext, are attracting increasing attention as potential candidates. This thesis argues that an understanding of the cognitive tasks involved in using these representations is fundamental to the subsequent design and implementation of usable tools for authoring and retrieving DR, and presents an analysis of the usability of a particular DR notation. The role of DR representations in group design deliberation, and issues relating to the introduction of DR into organisations are also briefly explored.

The *design space analysis* (DSA) approach to DR was studied, using the semi-formal *QOC* (Questions, Options and Criteria) notation. Software designers' use of QOC was analysed over a series of five studies, employing a video-based, observational methodology, and paper-based media. Four studies analysed QOC-use during design problem solving, one study addressed QOC-retrieval (from a simulated database), and the last study considered obstacles to the introduction of QOC in an industrial context.

A substantial and consistent body of evidence was gathered, describing the demands of the core representational tasks in using QOC, and the variety of strategies which designers adopt in generating and retrieving QOC. In addition, evidence was found suggesting that an argumentation-based design model based around laying out multiple Options is inappropriate during depth-first, 'evolutionary' modes of working, which centre around developing individual Options. Also reported are a range of requirements for future QOC tools, a summary of ways in which QOC supported design problem solving, two iterations in the design of a QOC tutorial, a 'styleguide' of characteristics for well-formed QOC structures, and several recommendations for extensions to the QOC notation.

QOC-use is considered from a number of conceptual viewpoints. Cognitive analyses which consider QOC-authoring as a form of knowledge elicitation, and of writing, point to the importance of rough, intermediate QOC representations to support early idea-structuring prior to explicit argumentation. Two further analyses draw on approaches developed elsewhere: DR systems are described and contrasted along several generic 'cognitive dimensions,' and the role of QOC representations in group deliberation is interpreted within a framework for shared workspace activity. Lastly, the thesis considers the tension between process-oriented, narrative DR and retrospective, rationalised DR. The data suggest that when the notation is unfamiliar, and/or ideas are undeveloped (as in initial DSA), the two approaches to DR in practice yield very similar results.

Contents

Figures

Tables

Acknowledgements

My thanks go to Rank Xerox Cambridge EuroPARC, the Science and Engineering Research Council, the Esprit AMODEUS project, and the Department of Psychology at York for funding and resourcing this research. Furthermore, whether I found myself north or south, the York HCI Group and EuroPARC provided me with stimulating environments in which to work, which made all the difference.

I would particularly like to thank Nick Hammond, my supervisor, for his consistent encouragement, advice, and backing, and Allan MacLean for his ever optimistic outlook and incisive feedback. Discussions with many other individuals helped broaden my often narrow perspectives, whom I thank. Of particular value was input from Victoria Bellotti, Jack Carroll, Thomas Green, Frank Halasz, Jintae Lee, Tom Moran, and Gary and Judy Olson. Thanks are also due to Tim Anderson, Adrian Smith and the Impact Project at Nestlé Rowntree in York, and Logica Cambridge for participating in the designer studies; also to David Elworthy for transcribing and discussing the Case Study 2 videos.

To Mum, Dad and Peter, I am indebted to you for the love and encouragement which you've provided, and for the regular stream of mail during writing-up, even though you rarely heard back from me.

To Eva, Angela, Catherine and Frank – you've been a strength beyond words. My 'Harrogate family' has been and will continue to be a source of emotional and spiritual, not to mention *physical* strength – are those Yorkshire puds in the oven Eva? • Angela Hulme, being able to write to you kept me sane last year • Chris T., you were solid, and there when you were needed most • Hilz, very simply, you're the maddest, most loveable nurse I know! • and Rich, it wouldn't have been the same without sharing the office, the tennis, the darts, and all the rest.

Thank you Warehouse (you even provided me with designers!). Watching the group grow, and crumble, and grow in fellowship, and seeing the vision, images and music take form and fuse is one of the most exciting and scary things I've ever done – and this is just the start. I'm beginning to understand what incarnation means.

Thanks Gavin, for living with me for two years... PhDs, Houses, Joy, The States, Jobs, Women, Bike Crashes, Cars... it's been good sharing it all with you.

And Jackie, you have been the best thing to happen to me. *Now* I know why they always say it – "thank you for your patience" – when producing this block of paper consumed the most sociable and unsociable of hours. Now *you* know why I didn't warn you ten months ago! Your love has been the most welcome distraction from work I could have asked for.

Above all, I acknowledge Jesus Christ as my Creator and Lord, under whom all else falls into place, whose love and guidance I have particularly known over the last three years.

Introduction and Overview of Thesis

0.1 MOTIVATION FOR THE RESEARCH

A design rationale (DR) is essentially a representation of the arguments behind design decisions. A DR should answer the question *Why...?* about the design it describes.

Research into DR has emerged only in the last few years as an identifiable field in its own right, as the result of several different conceptual and technological developments (reviewed in detail in Chapter 1). The domain in which DR is of most concern, and the focus of this thesis, is computer system design. Due to systems' complexity, and the still largely craft-based expertise of software engineering, software is relatively 'opaque' to somebody seeking to understand it, compared to better understood domains whose artifacts can be reverse-engineered more easily.

At present, the software design process is largely *artifact oriented*, that is, the emphasis is on generating and tracking the intermediate design artifacts – requirements, specifications, prototypes, user documentation – which lead up to the final system itself (Conklin, 1989; MacLean et al, 1991). However, the *process* of developing these artifacts remains largely implicit (e.g. hidden in minutes of meetings, or in designers' memories), and is consequently implicit, and often unrecoverable. Established design methodologies, whilst making the process explicit at one level, miss much of the reasoning which shapes design, as its exploratory, opportunistic nature cannot be modelled beforehand [see §1.4]. The variability and uniqueness inherent in design may be one reason why structured methodologies are often not followed systematically (Curtis et al, 1988).

The move to reify the deliberation process will assist the many parties who need to understand a design. In a large, often multi-party project, this communicative role is critical. Project managers, other designers, implementors, contractors, and later on, the software maintenance and support communities need to understand why decisions have been made. Maintenance accounts for a large proportion of system lifecycle costs (Balzer et al, 1983); Conklin (1989b) suggests that up to half the maintenance effort is in understanding the system (see also Wild and Maly, 1988), that is, in effectively reverse-engineering the design to recover the original rationale, in order to appreciate the implications of changes.

In the broadest sense, written minutes, old notes and sketches, early software prototypes, and audio/video recordings of meetings all embody DR. The general problem which DR research addresses is how to minimise the *cost* of creating representations of reasoning, and how to make that reasoning easily *recoverable* – two requirements which trade off against each other. As discussed shortly (Chapter 1), current research is focussing largely on representing DR as argument networks. Making the structure of design arguments explicit facilitates automated retrieval and editing, and allows arguments to be added incrementally through cumulative changes to the structure. It has also been hypothesised that use of such formalisms could augment the deliberation process itself. The background to DR, and detailed differences in how argumentation can be represented, are introduced in Chapter 3. Suffice at present to note that, typically, argumentation-based DR notations link subnetworks of *issues* (a sub-problem to be resolved), *alternatives* (possible solutions considered), and *arguments* in some form (which are used to justify the selection of an alternative). Graphically, these appear in a form similar to those shown in Figure 0.1.

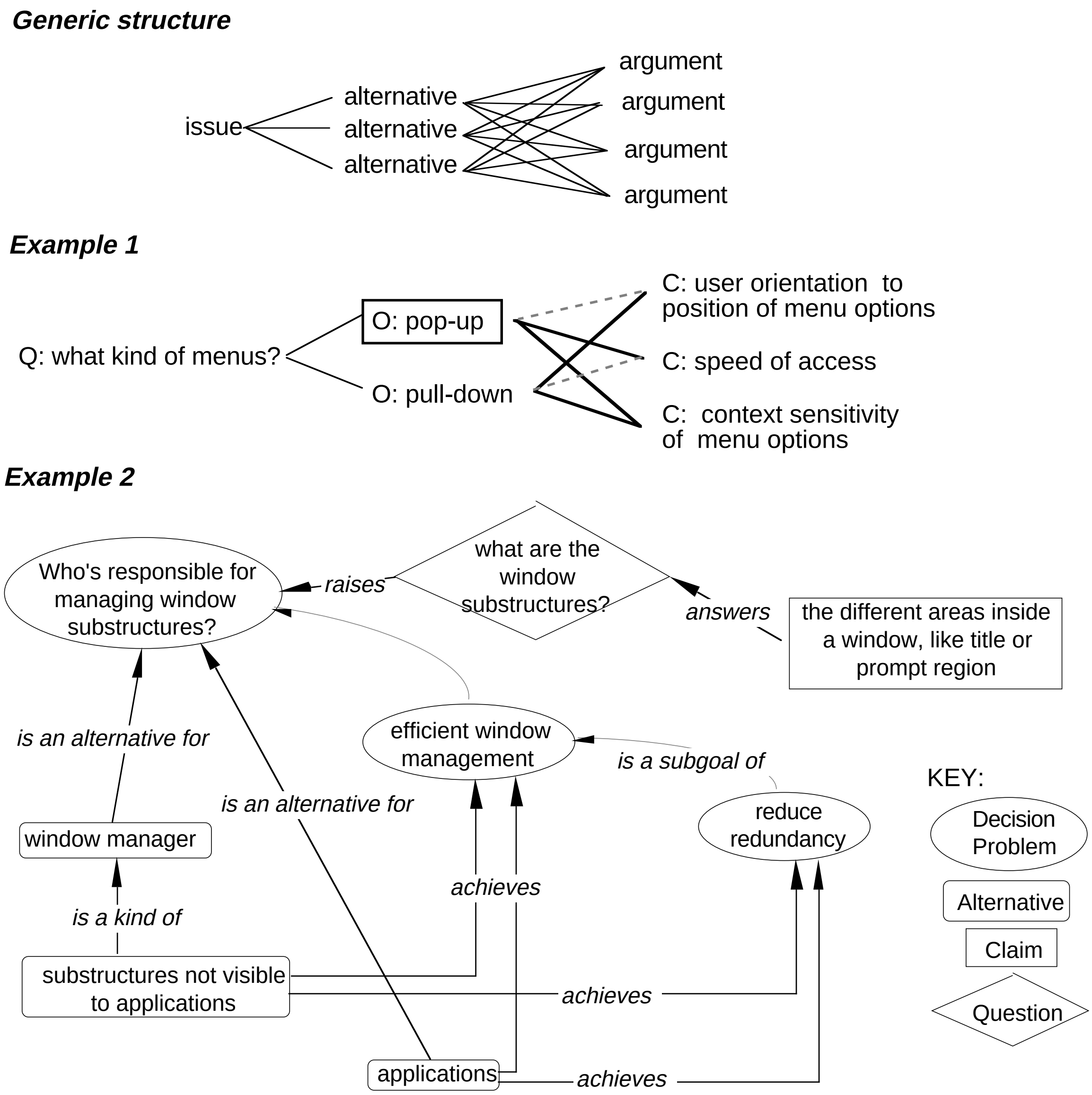


Figure 0.1: The core structure for an argumentation-based design rationale notation, and two examples using the notations QOC and DRL.

The relationships between entities are shown as links, which are assigned semantics depending on the particular notation (e.g. an argument can *object-to* an alternative), differentiated by labels or different styles of line. The kind of issues tackled also varies according to the approach to DR [§1.4].

In the following extract from a study of the software design process in a large organisation, the problem for DR is characterised as follows:

> Scheduling pressure was a major contributing factor at the point that many information capture systems broke down. There was typically a conflict between the desire to forge ahead with code and to slow down and document it correctly. In many cases, we were told that, "...if we could only have a tool to record our decisions...we could save weeks of rework."
>
> Yet, that same individual told us that, "...there is just no time... to stop to write things down." (Krasner et al, 1987, p. 58)

DR is admittedly one form of documentation. However, as the designer above suggests, the availability of a tool to capture decision-making *during* design deliberation would save reconstructing decisions long after they had been made, and possibly forgotten. However, as is also made clear, the introduction of new documentation overheads will not be popular with designers. The major task in DR research is to devise ways in which the benefits of DR can be reaped whilst minimising representational overheads which distract from the actual design work that needs to be done. Since *some* work is inevitably going to be needed, the goal is to turn the *process* of reifying DR to the designers' advantage, via the appropriate design of representational notations and support environments.

In order for DR authoring environments to be compatible with designers' ability to express ideas as DR, we need to understand the way in which representing DR interacts with the design process. If used seriously, it is almost inevitable that DR will be computationally represented, especially as semi-formal hypertext structures [§1.2]. However, as has been shown in other contexts (e.g. Monty and Moran, 1986; Tang 1989; Olson et al, 1990; Minneman, 1991), studying an activity in its 'natural' form (i.e. without computational support) is a powerful way to derive user and task-oriented requirements to guide subsequent technological intervention. An understanding of the activity which one hopes to support is fundamental to maintaining correct priorities in the subsequent design and implementation of usable tools.

The goal of this research, therefore, was to advance current understanding of the cognitive properties of DR formalisms from the designer's perspective, with a view to laying the empirical foundations for cognitive requirements for DR authoring environments. The wider dimensions of the design process such as social (e.g. Minneman, 1991) and organisational factors (e.g. Grudin, 1991) are not addressed in this research, although they undoubtedly play key roles in defining what will be ultimately acceptable in design practice. Within the context defined by social and organisational currents, however, designers must still perform cognitively complex tasks, and it is at this level that the cognitive analysis wields its power.

The general research question investigated might therefore be formulated as follows:

What does it mean for DR representations to be cognitively compatible?

There are two primary tasks in using DR, namely, authoring new DR, and retrieving existing DR for information or reuse in subsequent design. The studies in this thesis addressed both tasks (which cannot be divorced from each other), with an emphasis on the overheads of *authoring* accessible, reusable DR.

0.2 Methodological approach adopted in this research

It has been argued that design is still in a pre-theory stage (Finger and Dixon, 1989; Eder, 1990). The view that design research is still a long way from approaching the status of 'science' has some credibility. Tang (1989) notes that design research has moved significantly from attempts to *prescribe* design activity towards *describing* it. An interacting variable is whether one is interested in design at the level of cognitive, social, or organisational processes, as each requires the use of different investigative methodologies.

Given that design research is still in its formative stages, design *rationale* research is embryonic by comparison, with the added complication that attempts to understand the impact of introducing DR are confounded by our limited understanding of design 'in the natural.' It was thus necessary for this research, aimed at investigating DR usability issues, to adopt a methodology appropriate to this state of affairs. What is needed at present are considerable amounts of data on DR in use, with a view to characterising the psychological properties of the tasks of authoring and retrieving DR. With this as a basis, tools and methodologies can then be developed in a designer and task-centred fashion.

In the studies reported here, a *video-based, observational* methodology was adopted. The studies were essentially lab-based (with the exception of Case Study 3), with common design problems tackled between groups of designers (excepting part of Case Study 2). This was in contrast to simply recording uncontrolled design meetings 'in the wild,' since the specific aim was to gather data on the *cognitive* dimensions of DR notations in use. Video was used in order to gather detailed qualitative data, with the advantage that multiple analyses at different granularities and perspectives could be performed if necessary. As described in §6.1.2, however, quantitative analysis of DRs was also possible to some extent.

In total, six studies were conducted into DR usability, focussing on the QOC notation in particular. Studies 1-3 using experimental design problems, and Case Studies 1-3 studying designers using DR for their own work. A range of formats were used, varying in the number of designers, and the design problems tackled, as summarised in Table 0.1.[1]

	Number of designers working together			
	1	*2*	*3*	*>3*
own problem	Case Study 1	—	Case Study 2	Case Study 3*
artificial problem	Studies 1 & 2	Study 3	Case Study 2	—

*Table 0.1: Summary of the formats used in DR studies. *Organisational constraints prevented this format from being used in the end [Chapter 9].*

In summary, at present DR research is at the pre-theoretical stage, lacking basic data on the way in which designers use DR representations in design problem solving. In this context, to study cognitive issues in DR usability, there are conflicting interests between (i) gathering large amounts of data over as wide a range of conditions as possible, (ii) maintaining a sufficient degree of focus that useful data are gathered, and (iii) relinquishing experimental controls sufficiently to allow designers to work under ecologically sound conditions with respect to the variables of interest. It is asserted that an observational, but essentially lab-based methodology, centering on the rich data from video-recordings and the DR representations generated, constitutes an approach which is appropriate both in scope, and within the context of DR research as it currently stands.

0.3 Overview of thesis results

The research presented here is a combination of empirical data collection and analysis, plus more conceptually-driven analyses of DR authoring. The two developed in parallel, each informing the other. The format of the empirical studies is outlined above. The conceptual analyses explored several perspectives on DR usability: the parallel in cognitive tasks between DR authoring and writing [§1.3.2], an analysis of the position of DR notations within a conceptual space defined by generic 'cognitive dimensions' [§10.6], and an analysis of two contrasting ways in which DR can be used in group design [§10.4].

An overview of the thesis content is presented below.

Study 1 addressed issues associated with QOC retrieval. If a DR is to be accessible, what requirements does this place on its representation? What are the cognitive tasks in retrieving relevant DR in response to a query? How suited are graphical structures as a medium for DR retrieval, and how do users manage multiple representations of a DR when answering queries?

---

1 Note that a between-subjects experimental design was piloted for Study 3, but rejected due to methodological difficulties [§5.1.1].

Method: Seven subjects answered queries about the design of two systems by formulating requests to an imaginary hypertext DR tool to display specific QOC information. Subjects had available to them (for each system) representations of the QOC, plus a hierarchical "Criterion tree" showing inter-Criterion relationships, in both textual and graphical forms, and at different levels of detail.

Key results:
- a cognitive task model of the QOC retrieval process is developed as an analytic framework;
- errors in specifying QOC browsers are traced in the task model to different sources;
- multiple representations of the DR are necessary to support different kinds of queries; tools could compute virtual structures to map relations between them;
- Criterion trees which make inter-Criterion relationships explicit could be extended to compare the use of Criteria between different QOC DRs;
- the general conclusion is that the coherence of a DR's structure is critical to its reusability, but that this requirement is in tension with its semi-formality; this is particularly the case with DR approaches which focus on representing the deliberation *process*, rather than the structure of its *content*.

Study 2 and subsequent studies focussed on the DR authoring process. This study addressed a range of usability issues: what are the basic tasks involved in translating ideas into QOC? What difficulties are encountered in reusing current QOC, and in coherently integrating new QOC with old?

Method: Study 1 subjects were reused in this study. Without any prior experience in designing with QOC, they were asked to design a new feature for a system (from Study 1). The requirement was to maintain consistency with the current system (as represented in its existing QOC), and integrate the rationale for the new feature into the existing QOC structure.

Key results:
- different subjects use different principles to integrate the same idea into an existing DR; this is a property of semi-formal representations, but the absence of representational guidelines or constraints causes concern;
- it is not clear what counts as an Option or Criterion in QOC;
- difficulties are encountered in integrating QOC with actual design;
- there are early indications of the need for groundrules regarding QOC form;
- opportunities for tool support are identified.

Study 3 constitutes the largest study in the thesis, building on the initial explorations of Study 2. The aim was (i) to collect substantial amounts of data on the use of QOC, but after substantially more training than in Study 2, with a view to characterising in more detail the QOC authoring process and its demands, and (ii) to study the interaction between the extent to which ideas are developed, and the ease with which they can be represented as QOC.

Method: Twelve pairs of designers were trained in QOC, before tackling a design problem.

Key results:
❑ a clearer view emerges of the basic cognitive and representational tasks in using QOC;
❑ QOC authoring involves extensive revision of entities and structure;
❑ QOC must be able to express differential weightings of Criteria and Assessments;
❑ improvements which can be made to the QOC tutorial are identified.

The case studies represent a move towards testing some of DR's assumptions against designers working on real problems of a different nature from those studied in Studies 1-3. ***Case Study 1*** tracked a single designer over three sessions of using QOC (after training) on the project in which he was engaged.

Key results:
❑ QOC authoring patterns from earlier studies are largely replicated;
❑ QOC is found to be useful for understanding the relationships between ideas which have been thought about previously, and for choosing from several Options;
❑ severe difficulties are encountered in using QOC in an 'evolutionary' mode of working: problem difficult to structure into discrete Questions, the design is perceived and worked on as a single Option, and Criteria cannot be mapped to individual decisions.

The aim of ***Case Study 2*** was to evaluate a revised QOC tutorial, in particular an informal 'methodology' for developing QOC representations of the design space. This was taught and used by the designers in the form of a five-phase process model.

Method: The study used two established design teams in a software company, who were trained in QOC using a revised version of the Study 3 QOC tutorial. The process model was evaluated with respect to QOC use (i) for an exemplar design problem, and (ii) for a problem chosen by each design team from an ongoing project.

Key results:
❑ the designers engage in very little restructuring or revision of their DRs;
❑ QOC is criticised as being too cumbersome for design meetings;
❑ there is evidence that ill-structured problems are more suitable than well-structured ones for QOC-based argumentative design;
❑ more work is needed on formulating heuristics; in early stages of QOC use they tend to be forgotten, due, it is hypothesised, to the cognitive overheads of learning to use DR.

Case Study 3 set out initially to study QOC use within a small industrial project team. However, after the team had attended a QOC tutorial, organisational factors conspired to prevent its actual use. The data for this study were gathered from a subsequent interview with the project manager on prospects for the use of DR for software design within his work context.

Key results:

- although QOC was not actually represented, making Criteria explicit may have helped to structure problem analysis and meetings;
- QOC's potential as a medium for communication within a large project is particularly noted;
- problems are anticipated in the introduction of a new formalism in an international project – tables and lists may be more acceptable;
- organisational politics are one likely source of 'non-rational' Criteria.

General discussion in the final chapter considers the key themes which emerged from the studies. Firstly, it draws together results from the different studies which relate (i) to the observed benefits of using QOC during design, and (ii) to areas in which support could be provided by a DR environment. The main points for each are summarised below.

Observed benefits of QOC

- QOC acts as a concrete record of what has been discussed, enabling designers to review and regroup thoughts;
- QOC structure appears to encourage greater completeness in evaluating Options;
- spatial arrangement of QOC structures makes task-relevant information salient (missing Assessments; 'dangling' links);
- designers report that it is useful to have the issues clearly laid out for final decision-making;
- the process of constructing QOC helps to clarify vague, unarticulated ideas.

Requirements for a DR environment

The two key requirements identified are:

- awareness of the cognitive implications of expanding the notational vocabulary in order to increase expressiveness and computational power—do designers need, indeed, are they able to understand the vocabulary provided?
- support the whole process of rationalising argumentation, from initial idea organising to detailed argumentation—do not limit the scope of the tool to a semi-formal notation.

There were numerous instances in the studies where computational support would have been useful. Many of the facilities which third generation hypertext systems will provide would find application in supporting DR-related tasks.

Two ***conceptual approaches*** to DR use are considered in the final chapter, drawing from existing frameworks in HCI research.

- *Cognitive dimensions of DR*—DR systems are described and contrasted using generic ‘cognitive dimensions’ (Green, 1989). These focus the attention of the designer of DR notations and environments on generic properties of information structures which affect their usability.
- *DRs as shared representations in group design*—some of the data on QOC-use is viewed within a framework for workspace activity (Tang, 1991). This is used to characterise different roles which DR representations can play in group deliberation.

Lastly, three ***themes*** are reconsidered which recur in the course of the research, and deserve further investigation:

- *QOC methodology*—what are the prospects for systematising design space analysis such that the designer has more conceptual and representational ‘tools’ to hand when using QOC?
- *The notion of ‘good’ representational form in QOC*—a number of principles are formulated with respect to QOC’s main constructs, for developing well-structured representations which optimise (i) the analytical power of the approach for the designer using it, and (ii) the effectiveness of the QOC for subsequently communicating reasoning to others.
- *What kind of information should a DR preserve?*—there is a recurring tension in the discussion between the requirements of authoring historical, narrative DR which emphasises capture of the original design process, and retrospective, rationalised DR which emphasises abstracting out process details in order to make the logical content of the deliberation clearer.

Before DR was distinguishable as a focussed research field, Brown (1983) argued for the reification of the creative *process* as a key role which computers could play, both in enhancing the design, and in communicating design knowledge:

> ***Process versus product***
>
> By focussing in the design of empowering environments on the product of a creative effort, we are missing the real source of power for computer-based tools: the computer can record and represent the process underlying the created product. By making explicitly available to the user the series of steps and missteps that leads to the creation of a particular object or result, we create a basis on which to build extraordinarily powerful editing, merging, undoing, and transforming tools. Tools designed to manipulate this "historical" information, or *audit trail*, can be used to carry out intellectual and creative tasks of great complexity. [p.182, original emphasis]
>
> ... By maintaining an explicit audit trail of the steps that led to the created "object," not only can more powerful tools be constructed but, perhaps more importantly, the ability for one to understand what another has done is greatly enhanced. [p.183]

Design rationale research is now confronting the challenge of reifying the process without destroying it. This thesis explores the cognitive demands on the designer which are introduced in representing design reasoning as semi-formal argumentation, and considers ways in which the process can be supported through training, notational redesign, and computational environments.

Chapter 1

Design Rationale's Research Roots

1.1 INTRODUCTION

Design rationale research has emerged through the blending of several different research fields; the purpose of this chapter is to explore these 'roots.'

It was noted in the last chapter that DR can be embodied in many forms across different media. It is thus not surprising to find accounts of rationales in the research literature. Some examples of these are Johnson and Beach (1988), who pose and discuss the trade-offs to a series of questions relating to style-sheet design in document preparation systems; Woodmansee (1983) who describes the rationale behind decisions taken in designing a multitasking environment; Botterill (1982) who presents a user interface design rationale; McNall (1988)[1] reported that new members in a standards committee were required to "get up to speed" by reviewing the 'issue library'—forms documenting decisions taken in earlier meetings, and the reasons for and against them; lastly, Steele (1984) describes the Common LISP standard, using informal *rationale* notes throughout to explain the reasons underlying decisions in the language's design.

Although DR can take these different forms, it is the potential of representing arguments as semi-formal structures which has attracted most interest. The first part of this chapter identifies key ideas which have contributed to the use of computer-supported argumentation. Secondly, DR's roots in software engineering and design research are reviewed; the studies discussed here provide an empirical justification for DR, based on studies of design activity of different sorts.

The main sources from which current DR research draws are summarised in Figure 1.1.

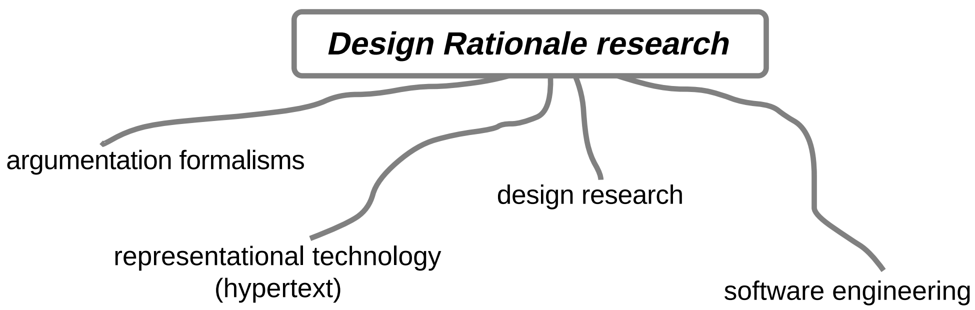


Figure 1.1: The 'research roots' of design rationale.

1 Personal communication cited in Conklin and Yakemovic (1991).

Understanding these roots sets the context for the DR argumentation notations and approaches surveyed in the next chapter. Before exploring these roots, however, the issue of representational technology is introduced. Hypertext tools have emerged as the dominant representational medium for DR for a number of reasons, and these are examined next in order to introduce concepts which arise throughout the rest of the chapter, and indeed, the thesis as a whole.

1.2 REPRESENTATIONAL TECHNOLOGY

The medium used to represent DR will play a critical role in determining whether it is accepted by designers. Whilst implementational issues are not the focus of this work, the study of DR authoring with pen and paper highlighted numerous requirements for computational DR environments [§10.7]; furthermore, understanding the task is crucial to deriving appropriate user interfaces for future tools. This section summarises the key features of the hypertext systems most commonly used for constructing and manipulating informal structures of the sort characteristic of design rationale argumentation.

1.2.1 Semi-formal notations and hypertext

Hypertext is a term coined by Nelson (1965), for a concept whose initial conception is generally credited to Bush (1945). Bush envisioned a system which would allow users to create and retrieve trails of relevant information, mimicking human associative memory; the essence of hypertext is in being able to link objects at will, and subsequently traverse those links.

Hypertext systems realise this by representing data as networks of *nodes* and *links*. Depending on the system, nodes can vary from basic text and graphics, to extracts of digitised animation, audio, and video. Links between nodes may be assigned *types*, reflecting the relationship between the nodes they connect, and nodes may also be typed to indicate for instance, the media it contains (a *text* or *video* node), or its functional role in the network (a *comment*; a *goal*; a *map*).

The user can navigate by 'jumping' from node to node by selecting (usually with a mouse) a screen-object (e.g. a word, picture, or icon) highlighted to show that it is one end of a link to another node or part of the document. The numerous cross-references in this thesis would take the reader directly to the relevant section if this was in hypertext. If the system provides an overview 'map' of the hypertext as an explicit network or hierarchy, often the map's nodes are direct links to the nodes they represent.

Obvious conceptual parallels with knowledge representation in artificial intelligence exist, and are now beginning to be exploited (Fikes, 1988; Barlow et al, 1989; Lee, 1990). A continuum exists, which describes how formally the structure and content of knowledge is represented (in terms of what is computationally tractable). At the 'informal' end is free text, such as this document. In this form, the system can 'know' very little about the semantics of the domain, and as such can offer very little computational assistance in manipulating the content. At the other end are knowledge-based systems in which nearly all of the knowledge is formalised.

Situated in the middle ground is hypertext, described as *semi-formal* (Figure 1.2). The *content* of nodes is generally informal (graphics; free text), and as such essentially non-computable (although see Malone et al, 1988, reviewed shortly). The *structure* is formal, and computable—link types can only be used between certain node types, giving the system a syntactic representation of domain semantics. Semi-formal systems strike a balance in the trade-off between computational tractability (structure), and human tractability or usability (content).

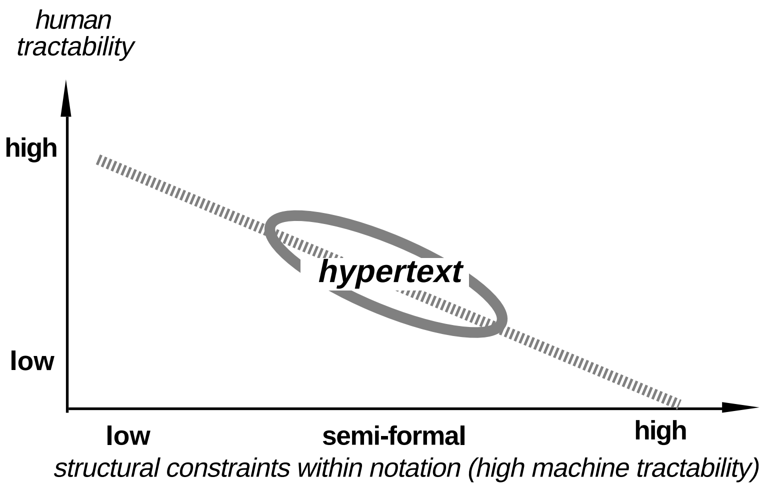


Figure 1.2: Semi-formal representations strike a balance in the trade-off between computational tractability, and usability.

Hypertext and expert-systems also differ in the important respect that the latter have an inference engine which actively processes information. However, as Halasz (1988) foresaw, the next generation of hypertext systems are beginning to provide more computational power through AI techniques. Whilst added power is of course desirable, the challenge is to preserve the usability of the tool.

1.2.2 Current hypertext systems

A comprehensive overview of basic hypertext concepts and systems can be found in Conklin (1987), and a shorter review in Halasz (1988). Halasz also discusses the functionality which third generation hypertext systems should possess (the first generation being systems from about 1965-1980, second generation from about 1980-present).

Bush's (1945) futuristic Memex, Nelson's (1981) Xanadu system, and others (e.g. ZOG–McCracken and Akscyn, 1984; Document Examiner–Walker, 1985; Hyperties–

Shneiderman and Morariu, 1986) have been concerned with making large amounts of existing material available as hypertext—the emphasis has been on *browsing* existing information. This is in contrast to work on exploring the use of hypertext to support the development of ideas in the first instance. From the perspective of DR, it is these *authoring* systems which are of interest as a means of representing and modifying the structure of design arguments. Table 1.1 summarises features in a typical second generation authoring system such as NoteCards (Halasz et al, 1987), gIBIS (Conklin and Begeman, 1989), Neptune (Delisle and Schwartz, 1986) or PlaneText (Gullichsen et al, 1986).

Feature	***Implementation***
Nodes	❏ Typed (text, graphics, animation...) implemented using a type hierarchy
Links	❏ binary, bi-directional ❏ labelled but not typed ❏ anchors can be whole nodes or points/regions within the node
Overviews	❏ browsers containing node/link diagrams of the hypertext structure ❏ hypertext structure editable via browser
Hierarchies	❏ special support for hierarchical networks
User interface	❏ multiple windows; graphics; mouse/menu driven
Extensibility	❏ programmer's interface (e.g. for predefining semi-structured node types)
Distribution	❏ single-user or multi-user central server with limited concurrency control
Versioning	❏ none
Storage	❏ standard files or relational database management system

Table 1.1: Architectural features of a typical second generation hypertext authoring system (adapted from Halasz, 1988)

Apart from the facilities described above, semi-structured systems display several other properties useful for representational work. Jordan et al (1989) describe several generalisable features for NoteCards which ease the overhead of generating and manipulating conceptual structures. One of these is *template* cards, master cards from which subsequent ones inherit properties such as text, graphics, fields, or predefined links to other cards. Another is *autolinks*: if the structure of a network is standardised, node type P may always be linked to nodes of type Q by a link type Z. This can be declared to the system, which then automatically creates a Q-node when the user adds a Z-link to a P-node. Autolinks can be set such that autolinks in connected cards automatically 'fire.' Consequently, hierarchical structures can be very quickly 'grown,' simply by creating one new link.

Malone et al (1988) describe their experiences with a hypertext system which makes extensive use of the sorts of templates described above. Nodes in Object Lens (Lai et al, 1988) are structured into forms with fields (e.g. author; subject; date; keywords); these attributes which are recognised by the system, increasing its power to display different structures. It is claimed that the extra computational services which can now be performed do not require excessive user effort. A facility of particular interest is the ability to declare rules which execute some action on nodes matching an attribute description. Using the attribute fields, one could declare, for instance, that all *Comments* about *Project X*, dated after *4th June*, should be automatically deleted.

Finally, Marshall et al (1991) present work on Aquanet, the successor to NoteCards, also heavily influenced by gIBIS. They specify "knowledge structuring tasks" as the target domain for support, describing a number of features aimed at assisting knowledge manipulation and presentation (focussing on argumentation as an example). These include specific object attributes devoted to its graphic appearance and spatial relationship to other entities, and support for automating and customising representational schemas (or notations). Aquanet *schemas* are used to define the syntax of the notation, and to some extent, enable it to be redefined in mid-discussion, via commands at a global level.

To summarise this section, the data model underlying hypertext makes it well-suited as an environment for representing entity-relationship structures. We now consider why such representations have come to be used so widely in DR.

1.3 REPRESENTING CONCEPTUAL STRUCTURES

The idea that DR should be represented semi-formally can be traced back to wider research into the development of computational support for reasoned discourse, or argumentation. Given that argumentation is a central activity in many key areas (theoretical research, design, writing, management, government), there is wide scope for representational schema (also described in the literature as argumentation formalisms, models, and notations) which could enable computers to assist in authoring, retrieving, evaluating, and modifying arguments.

A key assumption underlying argument representation research is that the structure of arguments can be analysed and represented independently from the content. All of the representational schemes to be described assume that argumentation can be represented within a vocabulary of statements and relationships with generic functional roles. Differences between notations lie in the constructs chosen to capture arguments, which determine the kinds of argument, and the level of detail which can be expressed.

In use, argumentation models are most commonly represented graphically as entity-relationship structures. One of the key hypotheses which prompted research into supporting argumentation was that by making the structure of arguments explicit, they can be more rigorously constructed and communicated (Brown, 1986; Smolensky et al, 1987), and these are still very much goals in DR research.

1.3.1 Representing argumentation structures

The work of four research endeavours has been selected to provide an overview of argumentation research which contribute elements seen in current thinking on DR. Briefly reviewed first, is the contribution of Englebart (1963), who although not dealing specifically with argumentation, was one of the first to envision the use of technology for manipulating what he referred to as "concept structures." Reviewed next is the early work of Toulmin (1958) on analysing and laying out the structure of arguments graphically. This is followed by work in the early seventies on argumentative approaches to design (Kunz and Rittel, 1970) and discussed lastly is the work at Xerox PARC in the mid-eighties, specifically the *Argnoter* tool for representing arguments in group design (Stefik et al, 1987).

1.3.1.1 "Better concept structures can be developed..."

Doug Englebart is widely recognised as a visionary in the history of interactive computing, having developed the mouse as an input device, multiple windowing, many of the current ideas about multimedia, wordprocessing, and one of the earliest hypertext systems. In his seminal paper, *A Conceptual Framework for the Augmentation of Man's Intellect*, Englebart (1963) foresaw the day when computers, far more powerful than available at the time of writing, would enable humans to overcome some of the limitations of their cognitive faculties by manipulating externalised 'concept structures':

> A concept structure ... is something that can be designed or modified, and a basic hypothesis of our study is that better concept structures can be developed—structures that when mapped into a human's mental structure will significantly improve his capability to comprehend and to find solutions within his complex-problem solving situations. (Englebart, 1963).

Englebart initiated his program for the 'Augmentation of Human Intellect' in 1968. The conceptual framework within which they were operating identified four *augmentation means*, defined as follows:

- ❑ *artifacts*—physical objects designed to provide for human comfort, or the manipulation of physical or conceptual objects (symbols);
- ❑ *language*—the way in which an individual models the external world; the symbols used to mentally manipulate those concepts;
- ❑ *methodology*—methods, procedures and strategies with which an individual organises his goal-centred (problem solving) activity;
- ❑ *training*—the conditioning needed by the individual to bring his skills in using augmentation means 1-3 to the point where they are operationally effective.

The H-LAM/T framework (Humans using Language, Artifacts, and Methodology, in which they are Trained) served as the conceptual basis on which the NLS (oN Line System) hypertext system was developed (Englebart and English, 1968). NLS was a precursor to much of the functionality in current hypertext tools, which as reviewed above, are the primary representational media for argumentation. The four-fold framework also serves to focus attention on the main issues addressed in this thesis, namely, the notations and representations of those notations used to represent DR, the way in which they are used in relation to design, and the necessary training needed to use DR notations.

In summary, Englebart foresaw the potential of computationally manipulated symbols as the means of achieving a symbiotic relationship between user and external representations in complex conceptual work, something which has remained the goal of much of the work reviewed in this chapter, and of DR research.

1.3.1.2 Toulmin's model of argument

The Uses of Argument by Stephen Toulmin (1958) was originally written as a challenge to the dominance in philosophy of formal, Aristotelian logic. Toulmin's aim was to develop a view of logic which was grounded in the study of reasoning practice. Taking argumentation as the most common form of practical everyday reasoning, he asked, "what, then, is involved in establishing conclusions by the production of arguments?" His analysis of the logical structure of arguments led to a graphical format for laying out the structure of arguments, a representational approach reflected in much subsequent argumentation work.

The notation consists of five components and four relationships (Figure 1.3). According to the analysis, an argument comprises (implicitly or explicitly), a fact or observation (a *Datum*), which via a logical step (a *Warrant*), allows one to make a consequent assertion (a *Claim*). The Warrant can be supported by a *Backing* if necessary (why the assumed Warrant is valid), and the Claim qualified with a *Rebuttal* (specifying exceptions to the rule).

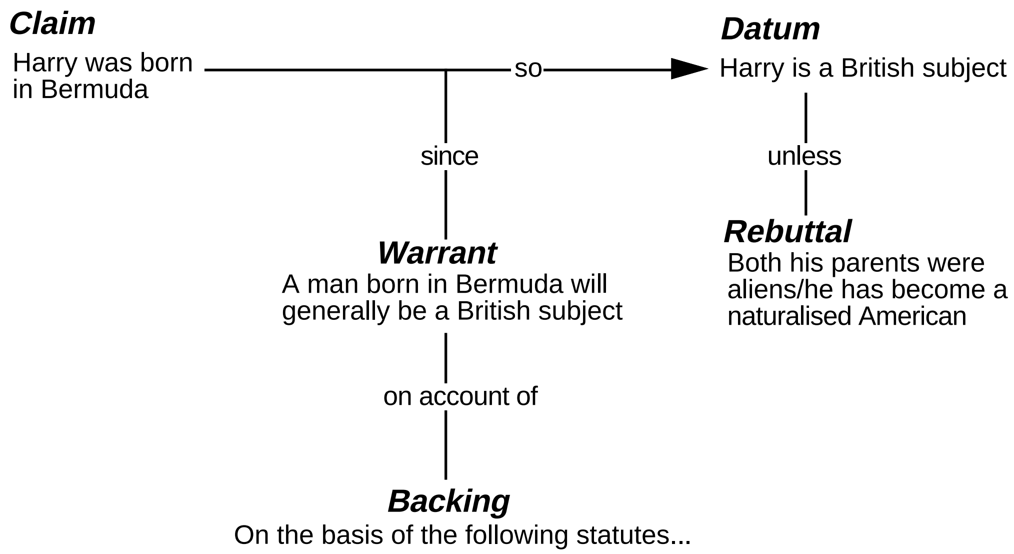


Figure 1.3: Toulmin's graphical argument structure

The purpose of this review is to set the conceptual background for the use of argumentation models for DR, so the considerable amount of debate which Toulmin's model stimulated is not of immediate relevance. Further reading can be found in Newman and Marshall (1990), and Hair and Lewis (1990) who set the model in context within the wider philosophical literature on reasoning. Further discussion of Toulmin can be found in Brockreide and Ehninger (1960), Cooley (1959), Cowan (1964), McCroskey (1965), Manicas (1966), and Burleson (1979).

One analysis of Toulmin which is however of direct relevance to the present discussion, is presented by Lee and Lai (1991b). They identify a set of argumentation 'spaces' which together constitute the domain of design rationale. This definition is then used to evaluate DR notations for expressiveness, that is, the extent to which each explicitly represents the different spaces. The spaces were identified as follows:

Argument space— all the arguments which relate to the design; entities in any of the other spaces can be argued about in the argument space.
Alternative space— the alternatives considered.
Evaluation space— the status of alternatives (e.g. 'rejected, accepted, in consideration'; or, on a different scale, 'very good, good, poor').
Criterion space— criteria relevant to the design; the basis on which evaluations are made.
Issue space— the above spaces relate to argumentation about a single issue; the issue space shows relationships between different issues.

Lee and Lai critique Toulmin structures on a number of counts. Because Toulmin has no notion of Goals, Alternatives, or evaluation measures, its scope is limited to the *argument space*, that is, in representing the arguments for and against the claims made in the other spaces. So for example, linking an Alternative to a Goal with a *supports* relation effectively makes a claim (A satisfies G), which can then be represented as a Toulmin Claim, and backed up or contested.

There are other representational weaknesses. One problem is that a Claim can only be *objected to* by supporting a Claim which negates the first; apart from being an awkward way to object to a Claim, there is no explicit relation expressing the mutual opposition between the Claims. Another difficulty is that the types assigned to nodes make them hard to reuse in different roles. A Claim can be reused as the Datum for the consequent link in an argument-chain; a Warrant may be a Claim from a previous argument, and so forth. A better solution is to place the semantic weight on the links, and use a more abstract node type which is context independent (as Lee's own notation, DRL, does [§2.1.6]). Streitz et al (1989) also use a hypertext representation of Toulmin structures (Figure 1.4).

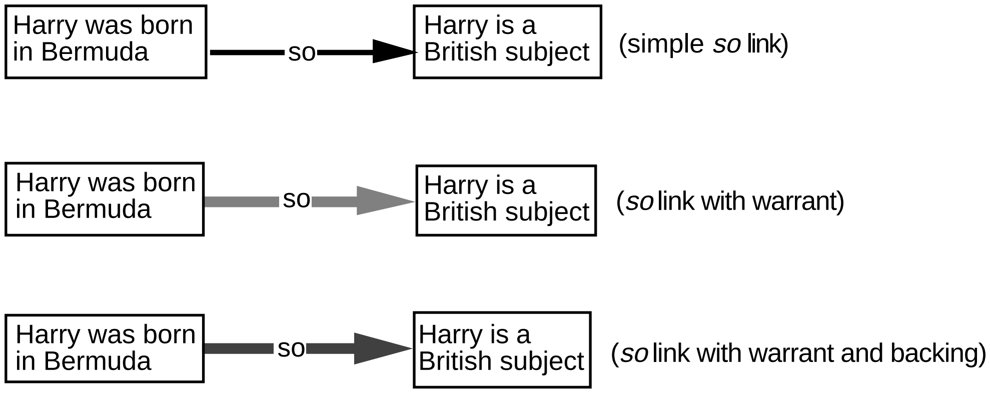


Figure 1.4: Alternative Toulmin representation (from Streitz et al, 1989). Nodes do not have functional roles embedded in their types, and Warrants are embedded in the link.

The key differences are the embedding of the Warrant and Backing in different *link types* (clicking on the link itself brings up a Warrant node), and the use of a common Statement *node type* in place of separate Claim and Datum nodes, releasing them to play different roles in different arguments.

In conclusion, Toulmin's work, particularly on using a semi-formal graphical representation, can be seen as a precursor to current notations. However, it suffers from a number of expressive weaknesses in the context of DR, which other notations remedy to varying degrees.

1.3.1.3 Wicked problems, the argumentative approach, and IBIS

The IBIS approach was developed primarily by Horst Rittel (Kunz and Rittel, 1970; Rittel, 1972). Other reviews of Rittel's philosophy and of IBIS can be found in descriptions of IBIS-based approaches to DR (McCall, 1986; Conklin and Begeman, 1989; Fischer et al, 1991), described shortly [Chapter 2].

The IBIS notation was developed out of Rittel's disillusion with the prevailing approach to design problem solving at the time. This emphasised the modelling and simulation of the problem domain, which in design manifested itself in attempts to automate design problem solving. He characterised problems such as those faced in government policy and design as 'wicked' (Rittel and Webber, 1973), in contrast to the 'tame' or 'benign' problems typically tackled in mathematics, chemistry, and engineering. The aim was not to belittle these disciplines, but to highlight the fundamental nature of problems which are hard to formulate, lack well-developed plans of action, and have no ultimate stopping rule or correct answer.

This class of design problem could not therefore be solved by formal models or methodologies, which Rittel classed as the 'first-generation' design methodologies. As an alternative, he proposed what he termed an *argumentative* approach to such problems (a second-generation design method). In the following extracts Rittel (1972) characterises the argumentative approach:

First generation methods seem to start once all the truly difficult questions have been dealt with already (...) The second generation deals with difficulties underlying what was taken as input for the methods of the first generation.

[Second generation] methods are characterised by a number of traits, one of them being that the design process is not considered to be a sequence of activities that are pretty well defined and that are carried through one after the other, like 'understand the problem, collect information, analyse information, synthesise, decide,' and so on...

My recommendation [for the future of design methodologies] would be to emphasise investigations into the understanding of designing as an argumentative process: where to begin to develop settings and rules and procedures for open-ending of such an argumentative process; how to understand designing as a counterplay of raising issues and dealing with them, which in turn raises new issues, and so on and so on.

[Argumentative design] means that the statements are systematically challenged in order to expose them to the viewpoints of the different sides, and the structure of the process becomes one of alternating steps on the micro-level; that means the generation of solution specifications towards end statements, and subjecting them to discussion of their pros and cons.

The argumentative approach led to the development of the IBIS notation as a medium through which open deliberation of issues could take place. The three key IBIS entities are Issues, Positions and Arguments, which can be linked by relationships like *supports, objects-to, replaces, temporal-successor-of, more-general-than,* and their converses. Figure 1.5 shows the basic working unit in IBIS, which grows into a network as more Issues are added and debated.

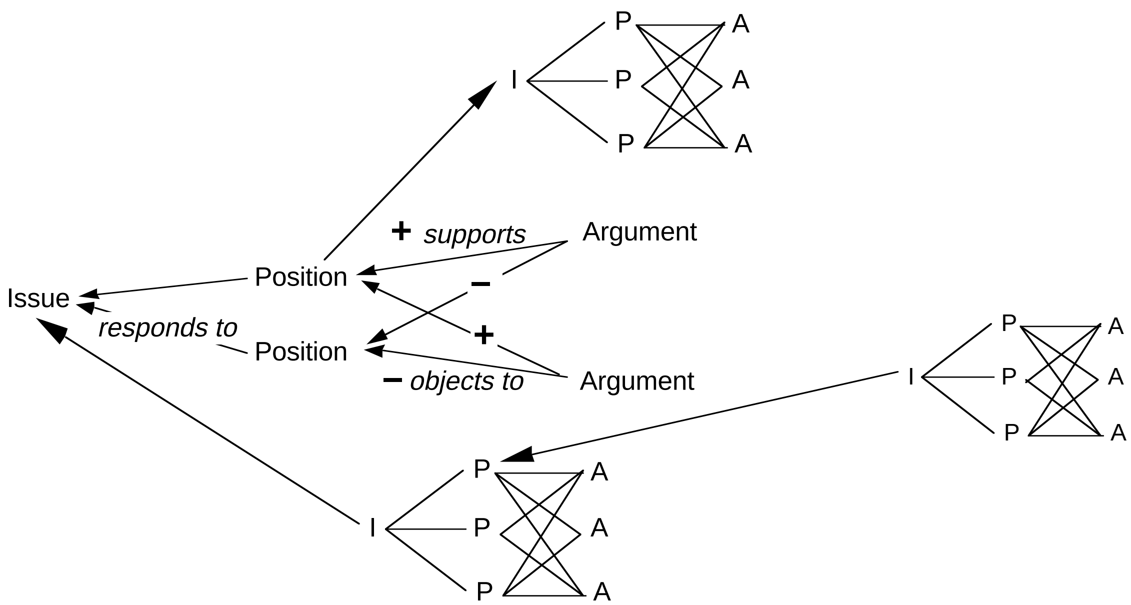


Figure 1.5: The basic IBIS structural unit of Issues, Positions and Arguments, developed to support the argumentative approach to design problem solving.

IBIS was used by a number of organisations for policy decision-making, including the United Nations, the West German Federal Office of the Environment, and the World

Health Organisation. Fischer et al (1991) report however that none of these attempts to implement IBIS developed beyond experimental prototypes, due to methodological and technological problems:

- the sheer complexity of an IBIS structure made it difficult to parse the links and substructures;
- there was no support for identifying the Issues which affected a particular decision – the Issue dependencies;
- there was no control over what kind of discussions took place – how should trivial issues be avoided, and the key issues identified and signalled?
- not every design question can be usefully deliberated, but IBIS deals only with those that can;

In sum, Rittel's work can be seen as establishing the bridge between the domains of design and argumentation. The argumentative approach to design elevated the importance of the *process* of understanding a problem from its minor status in first-generation design methods, to the central activity in wicked design problem solving. IBIS was the first explicit representation for reasoning in a design context—the first design rationale notation.

1.3.1.4 The Colab 'Argnoter' tool and its underlying philosophy

Described as "an argumentation spreadsheet for proposals" (Stefik et al, 1987), Argnoter was a tool to support the presentation and evaluation of competing proposals in design meetings. Argnoter is included in this review because both the thinking behind it, and its implementation as an experimental tool represent some of the earliest work on using modern workstations to mediate and manage explicit representations of design reasoning; the potential for computational support highlighted in the Argnoter work is now beginning to be realised in current DR research.

Research into computational support for idea structuring tasks like writing and argumentation was just beginning in the early 1980s. In a forward looking paper, Brown (1983 – quoted at the end of the introduction) focussed on the potential value of representing the *process* by which ideas and artifacts develop, as well as the final product. Brown pointed to the freedom from purely linear presentations of ideas which computers offered, and furthermore, discussed the potential of future tools which could manipulate and filter arguments, and the use of notations such as Toulmin. This was stated more fully subsequently (Brown, 1986), but the language used was still of future systems:

> Current communications tools and methods force the crafting of complex arguments into linear form for presentation, so that the web-like connections among ideas is hidden from view, making it difficult to see alternate interpretations and points of view. (...) As a result many of the underlying ideas, arguments and assumptions either remain implicit or are lost altogether. But consider the possibility of crafting new information tools to capture not just conclusions and the view of matters that supports them, but to

> allow the explicit representation of underlying assumptions and argument structures. (Brown, 1986, p. 484)

It was noted in particular that work was needed on developing *notations* with an appropriate vocabulary for the task domain:

> To accomplish these goals, we need a taxonomy of epistemological links for relating ideas, as well as link-related filters. That is, we must now think about giving users access to and utilisation of not just undifferentiated links, but links with appropriate kinds of labels. (p. 485)

Other work at the Xerox Palo Alto Research Center at this time was focussed on these goals, for instance the NoteCards generalised hypertext system for 'idea processing' (Halasz et al, 1987) and Marshall's (1987) work on representing Toulmin argument structures in NoteCards. The theme continues with the Aquanet hypertext system (Marshall et al, 1991). The Argnoter project described below was therefore only one of a number of projects linked to a research program which has played a formative role in the emergence of present DR research.

The Colab room was developed as an experimental meeting room for which a suite of tools was being developed to support different group tasks. Participants each had a workstation, and instead of a whiteboard, a large wall-display was used as the focus of attention for group representational activities. Stefik et al (1987) describe two tools under development, Cognoter and Argnoter. Cognoter was developed to assist the group process of developing a linear presentation of ideas, for example for a talk or research paper. The tool supported several phases of activity, from *brainstorming*, through *organising*, to *evaluation*, with different tool functionality for each phase. Lessons learnt in mediating discussion via Cognoter are documented by Tatar et al (1991).

Argnoter was developed to support the presentation and evaluation of proposals which had to some extent already been worked out (perhaps using Cognoter). The assumptions behind Argnoter were very similar to those underlying DR, namely, that

> [Design] is essentially a dialectic between goals and possibilities... in collaborative design tasks, this interaction and tension between goals and alternatives must play itself out in the communications among collaborators. (p. 38)

> A major theme of Argnoter design is that alternatives be made explicit: Proposals themselves are explicit, as are assumptions and evaluation criteria. (p. 38)

> A major working hypothesis behind the design of Argnoter is that making the structure of arguments explicit facilitates consensus by reducing uncommunicated differences. (p. 40)

Argnoter can therefore be regarded as a 'group DR' tool – a way to represent design arguments explicitly, but with the group process adding another dimension.

The emphasis on supporting *group work* led to an explicit model of the *process* which design meetings follow when making decisions, which was embodied in the tool's phases:

proposing, arguing and *evaluating*. Proposals are posted publicly on the wall-display as a textual description to which other representations may be linked if wished (e.g. sketches). 'Pro' or 'con' arguments are listed under a proposal, if necessary modifying previous proposals to create chains of arguments.

Finally, the evaluation phase is intended to examine the assumptions underlying the arguments. This is accomplished in Argnoter by commenting on the arguments which have been used, for example whether an argument was really believed or not. Stefik et al discuss the notion of 'belief sets' which are sets of arguments which share assumptions. This led them to the analogy of sets of 'argumentation spreadsheets,' each spreadsheet being a matrix of proposals against arguments sharing common beliefs. The goal was that the consequence for a proposal of changing an assumption would be seen immediately. The last step in evaluation was the selection and ranking of criteria. Agreeing on beliefs and on criteria were delineated as separate dimensions, and it was argued that the process of striving to agree on rankings and assumptions in Argnoter would help designers recognise where their differences lay.

Some of these ideas have been developed in current DR research, particularly by Lee (1990) who has focussed most on the scope for computational services for DR argument structures. Stefik et al's node types correspond to the generic DR entities of issue, alternative, and argument. In SIBYL (Lee's system) the move to make assumptions explicit translates into Questions and Claims about Alternatives, and the concept of belief sets displayed as tables is realised in SIBYL's Decision Matrices which update to reflect the strength of support an Alternative has in terms of supporting Claims.

To summarise, the research strategy exemplified in the Argnoter work is one of the earliest statements of a key tenet of current DR research – that computational support for making the structure of arguments explicit needs to be explored. The ideas described by Brown and Stefik et al were precursors to the more expressive DR notations and computational services which are now being developed.

1.3.2 Research into writing cognition, and writing-support tools

Whilst DR focusses on exploring and rationalising *design* reasoning, another area of 'idea processing' research has been in developing tools to support *writers* in organising their ideas into coherent texts. As described below, the problems which this research field is tackling overlaps significantly with those facing DR tool builders. Writing has been frequently described as a design task (Thomas and Carroll, 1979; Goel and Pirolli, 1989; Streitz et al, 1989), and the distinction between the fields is blurred even further with writing tools which support explicit argumentation, using notations like IBIS and Toulmin. As writing has been studied far more extensively than DR authoring, it is instructive to consider some of the support tools developed, especially those motivated by cognitive analyses of writing.[2]

Current research into writing-support tools is dominated by a concern to facilitate the smooth externalisation of internally represented structures, without constraining the author counterproductively. As the work reviewed below shows, the emphasis is on providing *multiple representations* of the ideas to be organised, covering the whole representational process from initial ideas to restructuring for different purposes (conceptual perspective; rhetorical strategy). It is accepted without question that authors recruit and construct different cognitive representations at different phases in the authoring process.

The problem faced by an author writing a document is very close to that faced by a designer wishing to represent reasoning. Consider the following quotes, on writing, hypertext authoring, and DR authoring, respectively:

> We will need tools for massaging, organizing, browsing something more akin to a stream of consciousness than a carefully thought out structure.
> (Brown, 1982, on requirements for a future writing environment)

> In early problem solving stages the enforcement of structure may get in the way.
> (Fischer, 1988, on current problems with hypertext)

> One common but subtle difficulty in hypertext systems is that it is sometimes unnatural to break one's thoughts into discrete units, particularly when the problem is not well understood and those thoughts are vague, confused, and shifting.
> ...the cognitive overhead of having to segment the 'muck' into discrete thoughts, identify their types, label them and link them is prohibitive.
> (Conklin and Begeman, 1989, on authoring with gIBIS)

It is not coincidental that one of the conclusions from this research [§10.7.3] is that well established elements in cognitive models of the writing process can be readily translated into requirements for a model of DR authoring. Let us now survey a representative sample of current writing-tools research.

Streitz et al (1989) describe a writer's tool which emphasises the mapping between cognitive models of the writing process and the tool's user interface and functionality. The user interface presents four windows in which to work, which are termed 'activity spaces,' corresponding to modes of idea structuring derived from models of writing. The spaces are *planning* (a meta-level for organising authoring), *content* (linked notes to be incorporated into the document), *argumentation* (Toulmin based organisation of arguments), and *rhetoric* (shaping the document to the audience and purpose of writing). The aim is to enable the writer to externalise ideas from several perspectives via one of the activity spaces.

---

2 For a broader overview of current research into computational support for writing see Williams and Holt (1989).

The rationale for supporting externalisation on which Streitz et al base their work, posits that different skills and additional knowledge can be brought to bear on *external* representations, which are impossible to apply to *internally* held representations (due to limited cognitive resources like working memory), such that external representations are open to modification and reinterpretation in more transparent ways than internal ones. A close relationship is drawn between writing and design, viewing the former as an example of the latter, as both involve constraint management. Their stated aim of creating an environment for "structured thought-dumping" could easily apply to DR. Schuler and Smith (1991) describe the implementation in more detail, which in particular uses PHI (McCall, 1986), an extension of IBIS, to organise arguments, and Toulmin to elaborate Position-Argument links.

Neuwirth and Kaufer (1989) describe a project aimed at elucidating "what makes a good external representation?" They begin by listing some of the problems which even skilled writers have (all of which could equally be applied to open-ended conceptual design):

- focus on details at the expense of larger goals, and distraction by irrelevant information;
- forgetting of useful information;
- searching for information for which there is only a partial specification;
- selection of incorrect paths leading to backtracking;
- losing track of goals.

Neuwirth and Kaufer go into more depth than many other writers in their cognitive analysis of representations to support synthesis writing. In the context of the cognitive architecture ACT* (Anderson, 1983), they present some criteria by which to evaluate writing representations, which are clearly generalisable:

- *encoding:* what is the number of elements encoded internally, which are relevant to the task, and how easy is it to operate on the internal representation produced by the external representation? (this differentiates between informational content and the cognitive tractability of the way it is presented);
- *storage and retrieval:* what is the likelihood that appropriate information will be chunked in, and retrieved from, declarative memory? This depends on how task-related information is grouped in the external representation. Retrieval as a chunk reduces the need for subsequent searches for related information;
- *controlling cognition:* how effectively can the user maintain current goals, the overall goal stack, and store intermediate results of operations?

To optimise the above criteria, different writing support representations are proposed for different tasks, which they contrast with those in other writing tools. For example, both Streitz et al's SEPIA system, and Smith et al's (1987) Writing Environment (WE) use a hierarchical tree to represent the document structure of sections; Neuwirth and Kaufer's

'SynthesisTree' is similar superficially, but it shows the *plan* for a written synthesis, not document structure. SEPIA and WE's representations are complementary; they would be of more use after having worked through tasks using Neuwirth and Kaufer's tools.

Neuwirth et al (1990) report the results of formal and informal evaluations of their PREP editor, for collaborative writing and commenting. It is interesting that they observe many writers using pen and paper for rough plans and drafts (e.g. of document structure), rather than online tools. These intermediate representations are then often discarded, their purpose having been served in orienting to, or agreeing on a perspective with co-authors. They note that the speed and flexibility of such 'scratch representations' may never be matched by online tools, but an understanding of their importance, together with the power of computational support and transformation, should encourage tool developers to support as much of the authoring process as possible.

Lastly, the Writer's Assistant (Sharples et al, 1989) is a writing support tool based on an analysis of the external representations utilised by writers (Sharples and Pemberton, 1988). In the resulting framework, they identified four main representations which authors use: unordered notes and idea-labels clustered spatially, non-linear structures like networks or spider diagrams, linear sentences or lists of ideas, and finally planar organisations, such as rough page mockups or formatted text. The tool was developed into a basic prototype system which supported most of the above requirements (it is intended that follow-on work will develop a new version).

In sum, examination of the writing research literature reveals a close parallel between tools to support the organisation of ideas in writing, and in design. There are differences of course, one being that in writing, generating the document is the final goal of the activity, whilst DR creation must be interweaved with creating other design artifacts such as sketches, specifications, and code. Another critical difference is that DR notations are semi-formal, not natural freeform text. Both of these, unfortunately, serve to complicate matters for the DR author. One is led to conclude that if there are problems in externalising and organising ideas as *natural language* in the production of a *document* (the final artifact), the *semi-formality* of DR notations together with their 'indirect' nature (they are a *representation* of the software artifact), serves only to widen the translation gulf a little further. Consequently, one of the goals of the research (Studies 2 and 3) was to explore the interaction between the extent to which ideas are developed, and the usability of semi-formal DR.

1.3.3 Empirical evaluations of argumentation tools and notations

Compared to the considerable literature describing writing and argumentation tools, there is a notable shortfall of empirical evaluations of these tools in use. Clearly, tools must be developed to a state where they are robust enough, and have sufficient functionality, that the underlying hypotheses can be usefully evaluated, but the danger is that features will continue to be added, even after the tool has reached a state where evaluations could begin. Given that the introduction of idea-structuring technology inevitably changes the task it sets out to support, the very models on which the tools are based may need to be modified, to accommodate the new modes of working which such tools engender.

Notwithstanding the disproportionately low number of evaluations, there are a number of reports of the use of argumentation schemas in a variety of domains and contexts. In some cases, they were embedded in tools, but in others, the representations were assessed independent of any implementation. However, given that usability problems with an argumentation *tool* will depend not only on the user interface and computational power available, but also on the adequacy of the argumentation *notation,* such analyses are valuable contributions to this literature. This section reviews the small number of reports of users' experiences in using explicit argumentation formalisms, summarised at the end in Table 1.2. Note that use of specifically *DR* notations is reported separately [§2.1.9]

Newman and Marshall (1990) discuss Toulmin's formalism in some detail after using it to represent legal arguments. Although some work was carried out in representing Toulmin with NoteCards (Marshall, 1987), Newman and Marshall's analysis focusses on the language rather than computational support. The main results of their analysis were (i) the need to extend the elements of Toulmin's vocabulary in scope; (ii) the formulation of four generic ways of linking Toulmin arguments; (iii) several recommendations for 'second order extensions' to the basic Toulmin argument including problem decomposition hierarchies, and matrices for comparing alternatives; and (iv) the importance of *learning* in the process of introducing representational tools – changes are required at the cognitive, social, and organisational levels in the way that work is conceptualised.

One of the few examples of argumentation tools benefitting real work is a relatively early report by VanLehn (1985). He describes how using NoteCards exposed two major flaws in his research. In one incident, by transforming a browser graph into a matrix, blank cells highlighted gaps in reasoning. In the second, a graph browser was computed over a set of ideas on notecards; the resultant structure showed disconnected subtrees (sets of issues) which should have been connected (as one theory). This led to an analysis of different inter-issue relationships, which in turn led to further insights.

He concluded that NoteCards' main advantage was its facility to "fool around with scratch organisations" in a way which paper-based media preclude. He also makes the important point that a NoteCards browser is driven by the contents of its constituent cards (the links embedded in the text of each card). To change the structure, the content of the nodes needs to be changed, so that it is impossible to compute a structure independent of the content of one's ideas. This decoupling was, however, exactly what had hidden the flaw in the argument up to that point—the neat hierarchical structure to the *document* he was creating was an independently imposed structure which finessed the *conceptual* weaknesses. In VanLehn's view, a NoteCards database *is* a theory of sorts.

Hair and Lewis (1990) report an interesting exercise, carried out as part of the EUCLID project[3] (Smolensky et al, 1988). Hair and Lewis selected two scientific papers, and represented the argument structures in Prolog, Toulmin, and EUCLID's ARL, the aim being to explore whether use of different argumentation schemes led to different – indeed *any* – insights into the arguments, or might help others grasp them.

The detailed results for each scheme are not covered here (though summarised in Table 1.2). Overall, however, Hair and Lewis felt that they could not attribute insights they gained into the arguments directly to any of the argumentation schemes, although the analysis required to use the representations helped clarify issues and relationships. (Of course, any investment of effort might have yielded the same benefits). They suggested that empirical studies were needed to show if different schemes lead to different understandings of arguments. They were also doubtful about whether presenting the arguments in any of the three formalisms would help somebody else grasp them better, as they were quite cryptic (Prolog especially), and there was a significant degree of subjectivity involved in constructing them. It was concluded that much work still needs to be done to make argumentation structures useful as a medium of communication.

1.3.3.1 Summary of research into argumentation tools and notations

The studies reviewed above are summarised below. Research specifically into DR is introduced in the next chapter, but is included below (in bold) to set it in context.

Domain and Tool	Representations	Evaluation of usability
RESEARCH THEORY		
(no tool) *Hair & Lewis, 1990*	Prolog	- many obvious steps have to be made explicit – cumbersome - very cryptic representation
(no tool) *Hair & Lewis, 1990*	Toulmin	- arbitrary distinctions need to be made between constructs - data, claim & warrant sufficed - graphical form helpful - impossible to explore alternative conclusions to the one modelled
EUCLID *Hair & Lewis, 1990*	Argument Representation Language	- simpler than Toulmin structures - less arbitrary judgements in structuring arguments
NoteCards *VanLehn, 1985*	own node/link types	- insight into weaknesses in theoretical reasoning
WRITING		

3 EUCLID is a prototype system based on a predicate calculus Argument Representation Language (ARL). It is part of a project to support theoretical reasoning of different sorts [§2.1.8.3].

Author's Argumentation Assistant (part of SEPIA) *Schuler & Smith, 1990*	issue hierarchy **PHI/IBIS** Toulmin rhetorical link types	none
tools for synthesis writing *Neuwirth & Kaufer, 1989*	notes summary graph synthesis grid synthesis tree	none (but in depth cognitive analysis of properties of representations)
SEPIA *Streitz et al, 1989*	issue hierarchy loose notes semantic net **Toulmin** combination of above	none
Writer's Assistant *Sharples & Pemberton, 1988* *Sharples et al, 1989*	loose idea-labels/notes networks of idea-labels/notes lists of idea-labels/linear text high level planar views of document	HyperCard screen mockups: - different views of document understood without trouble - new writing strategies would result from using the tool - navigational aids needed
Writing Environment *Smith et al, 1987*	network of titled nodes hierarchical tree editor view of a node linear text view of doct. zoomed in/out views of network or tree	none
DESIGN		
Design Space Analysis (no tool) *MacLean et al,1989; 1991*	**QOC**	(Shum – this research)
gIBIS *Conklin & Begeman,1989; Conklin & Yakemovic,1991*	**IBIS**	- industrial field study of itIBIS - shown to be useful in keeping track of decisions - training needed but quickly learnt - communication with outsiders using itIBIS difficult
IBIS *Kunz & Rittel, 1970*	**IBIS**	none
JANUS *Fischer et al, 1989; 1991* PHIDIAS *McCall et al, 1991*	**PHI/IBIS**	- report that early IBIS networks were too large; not enough control over content & level; IBIS not used successfully for policy-decisions - informal evaluations of PHI based tool showed PHI distracted designers from concrete design
SIBYL *Lee, 1990; Lee & Lai, 1991*	**Decision Representation Language**	none
rIBIS *Rein & Ellis, 1991*	**IBIS**	- difficult to use with little experience in IBIS method - complex user interface
LAW		
LEGALESE *Hair, 1990*	**IBIS-based: Issues, Facts, Arguments, & Laws**	- study of lawyers using pen & paper, or LEGALESE tool; independent judges - if notation was understood, arguments improved - written arguments were organised around the notation's model of argumentation, & judged superior
(some NoteCards work) *Marshall, 1987; 1989* *Newman & Marshall, 1990*	**Toulmin**	- extensions to Toulmin entity definitions required - additional structures needed, e.g. to chain arguments, decompose issues - usability of representational schemes depends on changes at cognitive, social and organisational levels

Table 1.2: Summary of studies into argumentation, writing, and design rationale (DR notations shown in bold)

This section has traced several sources of influence which have shaped, and in some cases continue to shape DR research activity. DR owes much to early visions of supporting problem solving by computationally manipulating externalised mental representations, a theme which has gained momentum with the surge in the computational power available to researchers over the last 20 years. The argumentative approach to design grew out of disillusion with first generation design methodologies, as did its semi-formal representation, although that can be traced back further to Toulmin. It is particularly noted that hypertext is playing an important role in making such formalisms manageable.

In the following section, a body of research is reviewed which clarifies the roots which DR research has in the software engineering and design research literature.

1.4 SOFTWARE ENGINEERING AND DESIGN PROBLEM SOLVING RESEARCH

Rather than providing a detailed overview of design research (either cognitive or methodological), this survey focusses on studies and theories of design which identify a role for DR. More general reviews of research into software design can be found in Tang (1989), Cross (1990), and Olson et al (1991). The aim is to show that a substantial body of research exists which identifies weaknesses in the design process and/or designers' cognitive resources which DR systems could assist in remedying.

The move away from the view that design could be prescriptively modelled or even automated has been documented already in the context of Rittel's work [§1.3.1.3], and so is not detailed again here. It is reiterated here as it was a significant step within the software development community towards a view of *design as dialectic*.

Parnas and Clements (1985) present the interesting argument firstly, that software design at present is not rational and that there will never be a perfectly rational software design process, but secondly, that it is worth 'faking' rationality in the design when presenting it to others, particularly in documentation. They present reasons why rationality will always be an idealisation of what actually happens, and the benefits which accrue from rationalising decisions afterwards. A feature of the approach is that documentation should record the design alternatives considered and rejected for later reference. This retrospective rationalisation of decisions is also one of the conceptual cornerstones in the Design Space Analysis approach to DR (MacLean et al, 1991), described in the next chapter.

Petersen (1987) argues that there are few good tools and processes to help assess decisions made during upstream (i.e. early, conceptual) design activities. He analyses the information management issues in using PlaneText, a hypertext system, to represent the issue base generated in a post hoc analysis of an existing design. He also briefly discusses the process of capturing issues *during* design using PlaneText, noting that the value of its facilities to provide views onto the issue base must be demonstrated, and secondly that the process of recording decisions must be unobtrusive.

Mostow (1985) provides an overview of ways in which artificial intelligence (AI) can support design, documenting a range of requirements which AI models of the design process must satisfy, and several promising avenues for research. One of these is making design rationale explicit in order to answer the sorts of "why?" questions which other designers are likely to ask. From the questions he uses as examples, Mostow seems to view DR primarily as a means by which AI design modellers can query why a particular model behaves in the way it does, for example:

why did the designer perform that transformation to solve that subgoal?

> Answer: to achieve its supergoal in the goal tree.

why did the designer choose one plan rather than another?

> Answer: a proof or other explanation of why that plan satisfies a given set of design criteria better than alternatives.

However, in his conception of DR, he also implicates questions of broader scope:

why did the designer use a particular set of criteria to compare alternatives?

> Answer: a proof or other explanation of why those criteria are appropriate given the trade-offs of the design space and the designer's preferences.

Whilst the above analyses were clearly developed in the context of software design, they are not based on empirical studies of designers. Let us now consider several studies based on data collected from reviewing or studying realistic design activity.

Rosson et al (1987) interviewed 22 designers of interactive systems in order to characterise current design practice. Apart from generating interesting data on designers' conceptions and practice in the areas of general design methodology, user interface design, and idea generating strategies, the study revealed that designers spent a lot of time in discussion with other people – colleagues, domain experts, and users. Rosson et al conclude, amongst a range of recommendations, that:

> ... one requirement for a design environment is a communication facility oriented toward explanation and rationalisation of ideas.

Clearly, DR in some form could be a representation well-suited to this role.

Schön (1983) has studied work in five professions (engineering, architecture, management, psychotherapy and town planning), from which he has developed a view of expert problem solving which characterises two primary modes of activity. One he calls *knowing-in-action*, the other *reflection-in-action*. Knowing-in-action is described as follows:

> I shall use *knowing-in-action* to refer to the sorts of knowledge we reveal in our intelligent action—publicly observable, physical performances like riding a bicycle and private operations like instant analysis of a balance sheet. In both cases, the knowing is *in* the action. We reveal it with our spontaneous, skillful execution of the performance; and we are characteristically unable to make it verbally explicit. (p. 25)

It is thus the cognitive state of an individual engaged in building the solution. It is unselfconscious, non-reflective, non-rational in nature, controlling action in relation to available artifacts.

Schön distinguishes between two kinds of reflective activity, which are prompted by something 'surprising' or unexpected occurring during work—a breakdown in knowing-in-action, rather like Heidegger's (1962) concept of breakdown when a tool is no longer "ready-to-hand" in the background, instead becoming "present-to-hand" in the user's consciousness.[4] The first kind of reflection is reflection-*on*-action, which results in a complete stop if engaged in during an ongoing activity:

> We may reflect-*on*-action, thinking back on what we have done in order to discover how our knowing-in-action may have contributed to an unexpected outcome. We may do so after the fact, in tranquillity, or we may pause in the midst of action to make ... a "stop-and-think." In either case, our reflection has no direct connection to present action. (p. 26)

Schön also identifies a second kind of reflection which serves to shape activity, because it is embedded *in* it:

> In an *action present*—a period of time, variable with the context, during which we can still make a difference to the situation at hand—our thinking serves to reshape what we are doing while we are doing it. I shall say, in cases like this, that we reflect-*in*-action. (p.26)

It is on reflecting-in-action which Schön mainly focusses, and its interplay with knowing-in-action. He asserts that these are mutually exclusive states of being in design, although the smooth switching between them in skilled performance can make it hard to differentiate the two. Reflection-in-action is the designer's response to how he perceives the current problem state "talk back" to him through his "reflective conversation" with the external world with which he is working.

Whilst Fischer et al (1991) adopt Schön's view of design as the conceptual basis for DR tool design [§2.1.4], it is sufficient in the present context to note that Schön's analysis of real-world design led him to a conception which assigns a central role to conscious rationalisation, and its interaction with other modes of working—if valid, this clearly has implications for attempts to introduce explicit argumentation into design.

Olson et al (1991) analysed ten design meetings from two organisations, conducted as part of a program intended to provide technology support for collaborative design. They

---

4 See Winograd and Flores (1986) for detailed application of Heidegger's ideas to human-computer interaction.

report that despite being drawn from a wide range of design contexts (different organisations, group size, experience, designs), consistent patterns of behaviour emerged, which enable them to begin to characterise group design. Their results are too wide ranging to detail here, covering problem solving strategies, use of knowledge, use of artifacts, and analysis of discussion topics. However, their analysis of transitions between activities during meetings is of considerable relevance to DR. They used the entities most commonly used in DR – issues, alternatives, and criteria[5] – as categories for classifying discussion. They found that design discussions were dominated by discussion of alternatives, criteria, and clarifications,[6] in both time spent on, and number of transitions from and to each category.

There is clearly an element of circularity in analysing discussions in terms of DR constructs, and then claiming support for DR by pointing to the content of the discussions. That being recognised, however, the large proportion of the meetings which could be classified under the three basic DR concepts lends support to the claim that DR notations in principle do not introduce fundamentally new concepts—indeed, that the entities *issue*, *alternative* and *argument*, are as close as they can be to the functional roles played by the majority of utterances in naturalistic design discussion.

Guindon et al (1987) present a cognitive analysis of 'breakdowns' in upstream design problem solving, performed by individuals. 'Breakdown' refers to ineffective design activity and its undesirable consequences, or complex activities which tax designers' limited cognitive resources. A functional cognitive model based on their observations was used to describe the internal and external representations of the problem and solution, inferred from behaviour, or actually observed. The main sources of breakdown in design were:

Breakdowns due to limited knowledge
- ❑ lack of specialised design schemas
- ❑ lack of a meta-schema about the design process (leading to poor resource allocation to different design activities)
- ❑ poor prioritisation of issues leading to poor selection between alternative solutions

Breakdowns due to cognitive limitations
- ❑ difficulty in monitoring all of the stated or inferred constraints
- ❑ difficulty in keeping track of all the steps or test cases during evaluative simulations of solutions

Breakdowns due to both limited knowledge and cognitive resources

---

5 Issues were defined as questions, problems, or aspects of the design addressed; alternatives were solutions or proposals about aspects of the design; criteria were reasons, arguments, or opinions used to evaluate an alternative. Details of the coding are procedure are discussed in the original paper.

6 A clarification covers questions asked, repetitions, and explanations to clear up misunderstandings.

- ❑ difficulty in tracking and revisiting subproblems where solution refinements have been postponed
- ❑ difficulty in integrating solutions to subproblems into a complete solution
- ❑ premature commitment to an initial solution kernel based on a priori criteria

Previous studies of software design reported the adoption of a primarily top-down process, with occasional deviations (Adelson and Soloway, 1984; 1985; Jeffries et al, 1981). In the course of Guindon et al's study, it became clear that the mode of design adopted by the designers was overridingly *opportunistic,* with less top-down decomposition, a theme which Guindon (1990) focussed on:

> ... opportunistic design is design in which interim decisions can lead to subsequent decisions at various levels of abstraction in the solution decomposition. (p.336)
>
> Opportunistic design is characterised by on-line changes in high-level goals and plans as a result of inferences and additions of new requirements. In particular, designers try to make the most effective use of newly inferred requirements, or the sudden discovery of partial solutions, and modify their goals and plans accordingly. (p. 337)

Thus, in upstream design, until the proper design decomposition is discovered, top-down analysis gives way to opportunistic exploration. Other research into design confirms the dominance of opportunistic modes of working with upstream, ill-structured problems (e.g. Olson et al, in press; Ullman et al, 1987; Visser, 1987; Green et al, 1987; Carroll and Rosson, 1985; Siddiqi, 1985). The substantial evidence for this mode of design confirms Rittel's views (1972) that all of the hard work had been done by the time 'wicked' problems had been worked into a usable form for first-generation design methods; Simon (1973) made a similar point in his analysis of ill-structured problems, commenting that,

> ...there is merit to the claim that much problem solving effort is directed at structuring problems, and only a fraction of it at solving problems once they are structured. (p. 187)

Guindon notes that whilst hierarchical goal decomposition and planning models like ACT* (Anderson, 1983) are still able to account for opportunistic behaviour, the implications for support environments are different. In tasks as complex as design, it is extremely difficult to make predictions with models. Guindon does however recommend further studies into the effect of the structuredness of problems, and suggests that tool environments built around different cognitive models might lead to insights; the problem is however that enforcing an environment on design changes the process being studied.

A number of recommendations for tools to alleviate the sources of breakdown listed above are made by Guindon et al (1987). Explicit DR is identified as potentially useful in several cases:

Issue prioritisation

- ❑ explicit DR should be used to structure issues relating to the domain, class of system being designed, and the design process itself.

Premature commitment to an initial solution based on a priori criteria

❑ explicitly represented argumentation could encourage exploration of alternative solutions and design processes.

Difficulty in expanding or merging reduced or partial solutions

❑ easily retrievable DR, linked to different alternative designs, would minimise loss of the initial motivation of the problem decomposition.

Furthermore, in the light of the specifically *opportunistic* nature of design, Guindon (1990) specifies the following requirements for tools:

❑ the tool environment should support rapid access and shifts between tools to manipulate different kinds and representations of objects: these include issues and criteria about the system and design process, design decisions expressed in a formal or semi-formal notation, and design process goal management.

❑ representation languages in the environment should support a smooth progression from requirements expressed informally, to design decisions expressed formally or semi-formally, to code.

❑ there should be easy editing and reorganisation of the requirements, design issues, and decisions, as incompleteness and ambiguity in the problem specification is reduced.

Lastly, Goel and Pirolli (1989) present an analysis of design cognition, as they attempt to articulate, within the framework of Newell and Simon's (1972) Information Processing Theory, distinguishing properties of 'generic design' as opposed to other kinds of problem solving. They delineate eight characteristics of design problems which are major invariants in the *Design* Problem Space (DPS), following Newell and Simon's notion of the Problem Space. They develop the DPS in close conjunction with their analysis of the Design Task Environment, which constrains problem solving in particular ways (e.g. many degrees of freedom, delayed feedback on the effect of decisions, there are no right answers, problems tend to be large and complex). The results from studies of three different kinds of design (architecture, mechanical engineering, and instructional design) are used to support their claims of invariants in the DPS. There are several facets to this analysis which provide 'hooks' for DR—DR notations may be able to support several of the DPS invariants which Goel and Pirolli identify:

(i) *The many degrees of freedom in design problem solving entail extensive problem structuring*: DR goal and issue hierarchies could assist in reifying the subproblem relationships (PHI's Issue-hierarchies are intended to support this specifically [§2.1.3]).

(ii) *The input to design is often in the form of goal statements – levels of goals (and also commitments) are mediated by abstraction hierarchies*: see (i)

(iii) *There is a tension between having to model (cognitively or otherwise) alternative design paths, yet make commitments*: laying out the design space and rationale for different paths should assist in delaying commitment whilst maintaining an overview of the contingencies, particularly with computational support which allows 'what-if?' exploration. In particular, the emphasis in DSA on not necessarily recording decisions in the order in which they occur, and delaying decision-making until the space is laid out could prove to be a useful strategy.

(iv) *Solution decomposition is not often complete – modules are 'leaky'*: MacLean et al (1991) note that Goel and Pirolli's 'leaky modules' are equivalent to dependencies between different parts of the design space. With computational support, these can be monitored.

(v) *Artificial symbol systems are necessitated by the limitations of our cognitive resources; the representations used have consequences for the above features of problem solving*: some approaches to DR aim not only to support retrieval of DR, but to augment design problem solving in different ways (DSA, and Fischer et al's work on integrated design environments in particular). As summarised at the end [§10.2], several beneficial properties of QOC representations and the *process* of creating those representations were observed.

Taken together, the studies reviewed in this section (i) point to weaknesses in current software design practice which could be supported by DR systems; (ii) direct attention to limitations in cognitive resources which constrain design problem solving—explicit DR could relieve some of the representational load; and (iii) provide us with an empirically sound basis for targetting the kind of conceptual, opportunistic design studied in the thesis.

Having set DR research in its historical and conceptual context, the next chapter introduces current research efforts exploring different ways in which DR can be represented notationally, and supported computationally.

Chapter 2

Approaches to Representing Design Rationale

This chapter reviews current approaches to and representations for design rationale. For detailed presentations of several of the approaches described here see the Special Issue on Design Rationale of the journal *Human-Computer Interaction* (HCI, 1991). Also recommended is the forthcoming book (Moran and Carroll, in preparation, 1992), which reprints these journal articles together with several other chapters to provide a broader overview of the state of DR research and thinking.

2.1 Different approaches and notations

The key features of the different approaches and notations are first briefly described, together with related work of relevance to DR. The order in which the approaches to DR are covered reflects very approximately the move from narrative to retrospective DR (a distinction introduced shortly), starting with IBIS-based work. Work related to DR is noted, and reports of the different DR approaches in use reviewed. §2.2 then identifies some of the tensions which arise from differences in the approaches. This sets an agenda of research issues, some of which are addressed by the work reported in this thesis.

2.1.1 GRAPHICAL ISSUE BASED INFORMATION SYSTEM (gIBIS): Capturing the design *process*

The work on gIBIS[1] has adapted Rittel's Issue Based Information Systems by extending its vocabulary and adding a graphical representation – each IBIS is displayed as a directed graph (Conklin and Begeman, 1989). Issues, Positions and Arguments are the main entities used, plus a set of relationships. Figure 2.1 shows the gIBIS vocabulary, and an example.

1 (pronounced "gibbiss")

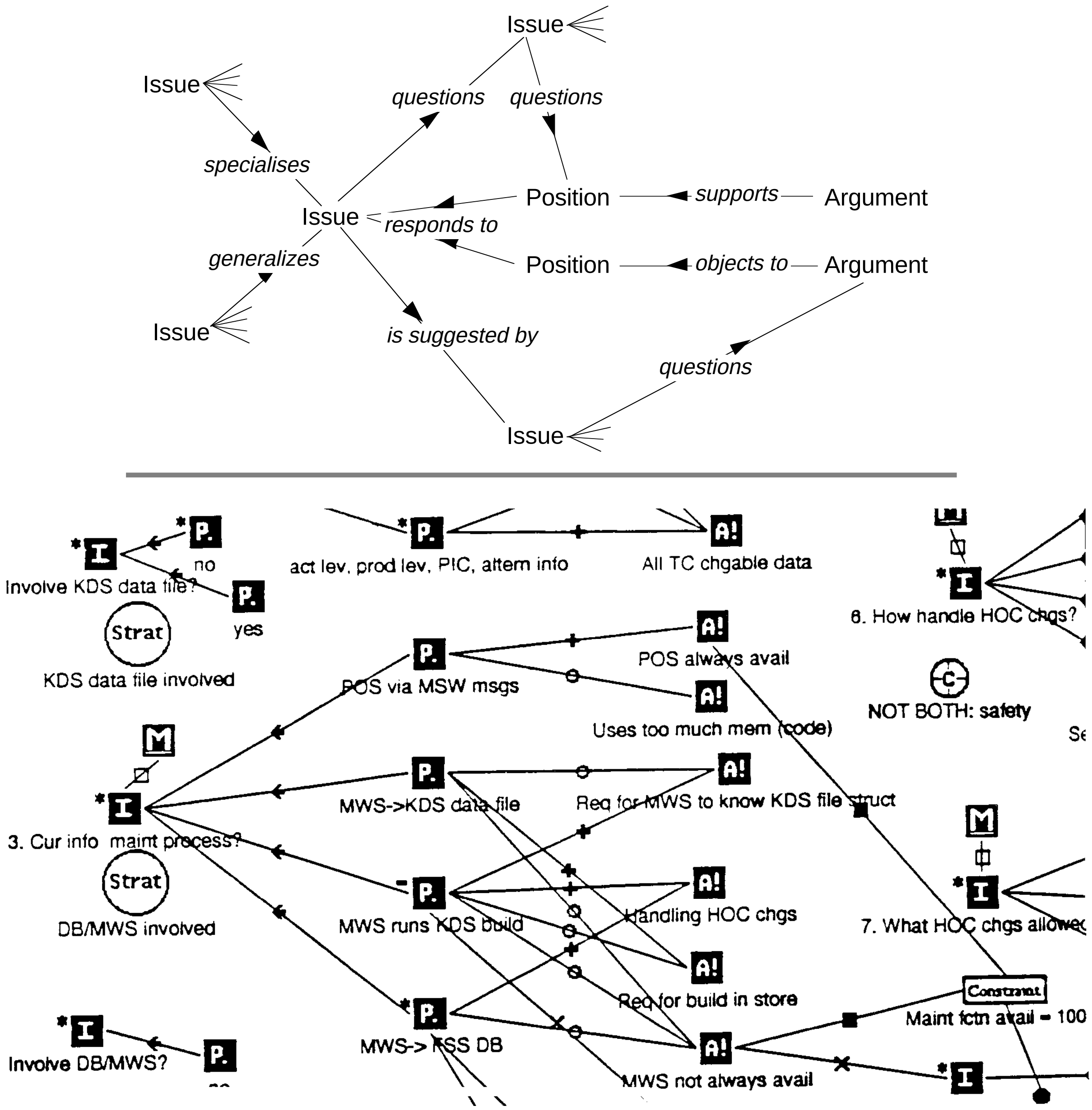


Figure 2.1: The gIBIS vocabulary, and an example taken from the tool's display. (Note that the example incorporates some experimental constructs such as *Composite* nodes (representing a cluster of nodes), *Constraint* nodes (expressing different interdependencies), and *Strategy* nodes (linking Positions sharing common goals or assumptions).

Conklin and Yakemovic (1991) identify a fundamental trade-off which approaches to DR must tackle, between the cost of DR authoring, and the cost in accessing and using the DR once authored. They give the two extreme examples of video recordings of design meetings (low creation cost/high reuse cost) and expert system knowledge bases for automated design (high creation cost/low cost reuse).

What Conklin and Yakemovic label *structure*-oriented DR, is typified by MacLean et al's (1991) approach [§2.1.5], which involves working on designing a "knowledge representation of the design space" which captures more than one potential design. The use of gIBIS they characterise as *process*-oriented. This places an emphasis on:

- use *during* design meetings – retrospective rationalisation is an overhead to be avoided;
- use of IBIS to *structure* meetings, so that participants make "rhetorical moves" within the notation, again, with the aim of minimising additional representational overheads;
- *capturing* the process of the meeting, with all of its false starts, and rejected alternatives, *preserving the order* in which they occurred;
- tracking the process by which a *particular* design evolves – the DR's reusability in other contexts is a secondary concern.

The gIBIS tool is a hypertext application running on graphics workstations. An IBIS is represented as a network, with other windows for displaying the contents of nodes, and navigation. Rein and Ellis (1991) have described the implementation of rIBIS (real-time IBIS), a group version of gIBIS for the collaborative development of IBIS structures during meetings [§2.1.9].

2.1.2 TAILORING IBIS TO SPECIFIC DESIGN METHODOLOGIES: Making DR's relationship to software engineering more explicit

Potts and Bruns (1988) present a model for relating entities in existing software engineering methods to IBIS-based design deliberation. In their model, a design history is made up by the network of intermediate *artifacts* produced en route to a finished design, artifacts being specifications or design documents, which are *derived* from one another through *deliberation nodes* (represented as Issues, Alternatives, and Justifications). The particular artifacts depend on the software method being supported. Based on IBIS, Issues derive Alternatives, which derive justifications. The key difference from other DR representations, namely the integration with software engineering methods, is achieved through deriving Artifacts from Alternatives. The syntax of the notation is summarised in Figure 2.2.

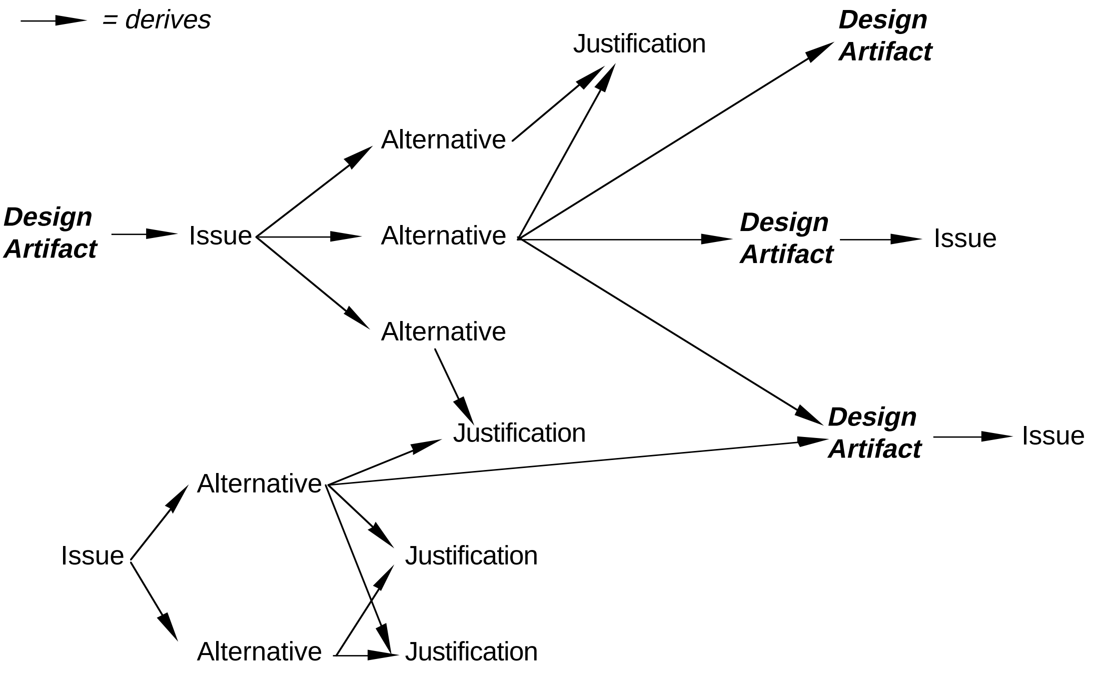


Figure 2.2: Potts and Bruns representation for integrating design deliberation with design process artifacts.

It is proposed that this generic model should be tailored to support specific design methods. Potts and Bruns present a worked example of the design of a text-formatter using the Liskov and Guttag method (Liskov and Guttag, 1986). A new entity specific to this method is the *Task*, anything performed by a procedural abstraction, which is incorporated into the IBIS method by raising a new Issue. Potts and Bruns' generic *Artifacts* are refined into Liskov and Guttag's *procedural specifications*, and *data abstraction specifications* and *Issues* into *behavioural issues* and *encapsulation issues*. Figure 2.3 reproduces part of the analysis to illustrate the adaptation to Liskov and Guttag. Understanding the model in detail is not necessary; it should simply be understood that the generic model's entities are refined to accommodate a particular design method's vocabulary for deriving new artifacts.

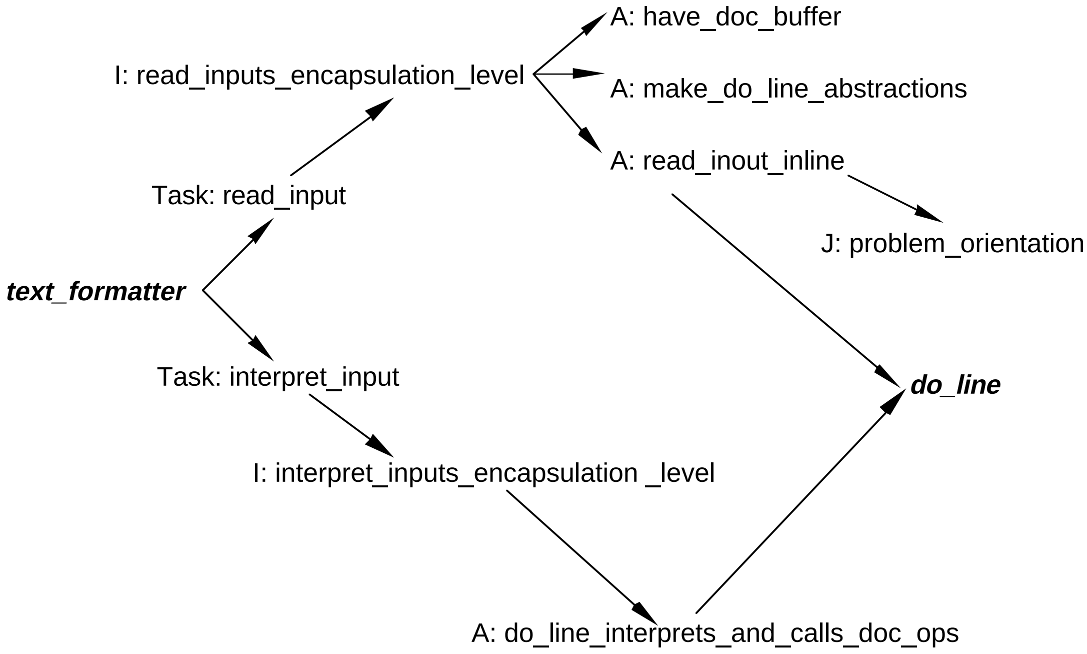


Figure 2.3: Example of Potts and Bruns IBIS-based notation tailored to the Liskov and Guttag design method.
Key: *Artifacts* in italics; I=*Issue*; A=*Argument*; J=*Justification*)

The transformations of Artifacts through deliberation was represented by Potts and Bruns using the PlaneText generalised hypertext system (Gullichsen et al, 1986), whose structure was verifiable by Prolog rules which 'knew' the Liskov and Guttag syntax.

Potts and Bruns touched on two issues which have since emerged as key questions in DR research:

- the overheads of authoring the design history – the tool must be non-intrusive; the names must be succinct yet informative (results reported later confirm that naming can cause significant problems [§6.2.3]);
- what kind of history is desirable? – they wonder if creating design history of this sort concurrently is too disruptive to design activity, and whether the DR should be retrospective.

2.1.3 Procedural Hierarchy of Issues (PHI): Structuring IBIS argumentation around the relevant Issues

The PHI method was developed by McCall (1979; 1991) to augment IBIS by overcoming some of its evident limitations since first being proposed by Rittel. PHI expands the definition of *Issue* and introduces a new set of inter-Issue relationships.

An Issue in IBIS was simply a design question which was deliberated, whereas a PHI Issue is any issue whatsoever. PHI Issues are related by *serves* links, such that Issue A *serves* Issue B if resolving A help to resolve B. The *serves* relation can be expressed as A *is-a-subissue-of* B, if B was raised first, or as A *is-an-antecedent* of B, if A was raised first. The two are logically equivalent, however, and appear to have been introduced simply to accommodate process factors (i.e. the order in which ideas arise). A PHI structure is quasi-hierarchical (an Issue can have more than one parent), and has generally been represented as a textual outline, with subIssues indented. PHI Issues are resolved by argumentation in the usual IBIS style, or simply by decomposition. The prime benefit claimed for PHI is that it focusses deliberation on Issues which serve the stated aims of the design (i.e. the top Issue). If a problem cannot be shown to serve part of the hierarchy, then it is likely to be irrelevant.

PHI has been implemented in several tools. MICROPLIS (McCall et al, 1981) was the first issue-based hypertext system to be developed anywhere; this was substantially extended to include a CAD component in JANUS (see next section), which has been developed further into PHIDIAS (McCall et al, 1990). In all of these, the PHI argumentation is presented in textual outline form.

2.1.4 JANUS AND PHIDIAS: Linking design representations to design rationale

Work based at Colorado University (Fischer et al, 1989, 1991; Lemke, 1990; McCall et al, 1990) has focussed on integrating design argumentation with what they call design ‘construction,’ that is, representations of the design (most work has used CAD layouts for kitchen design). The JANUS system monitors CAD layouts as they develop, and rule-based design ‘critics’ warn the designer if a design guideline is violated; the designer can then request PHI argumentation to explain the rule. PHIDIAS supports structural-searches of the issue-base, hides issues if previous decisions make them irrelevant, and allows relevant argumentation to a CAD object to be displayed simply by selecting the object. Whilst PHIDIAS does not use the rule-based critics of JANUS, it demonstrates how design artifacts can be linked to design knowledge which cannot be easily formalised as rules. A screen from PHIDIAS is shown in Figure 2.4.

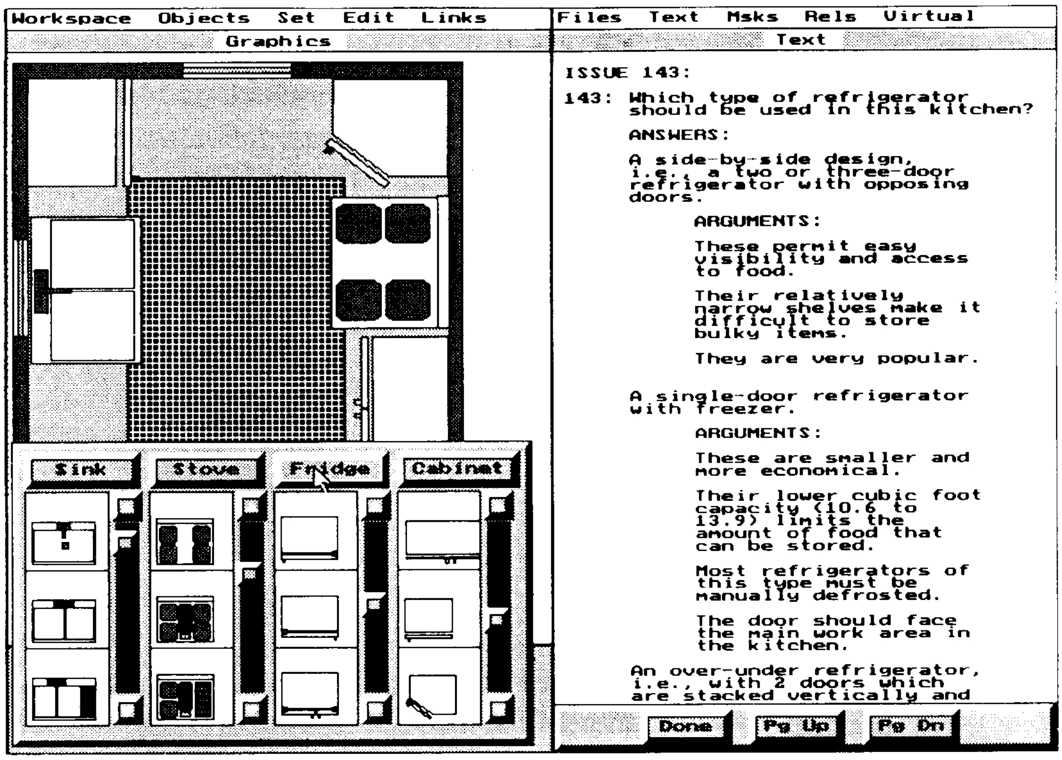


Figure 2.4: A screen from PHIDIAS showing CAD elements and argumentation associated with the selected object ('Fridge').

As noted in the last chapter, Fischer et al have drawn on Schön's theory of design activity as the conceptual basis for JANUS. Knowing-in-action and reflecting-in-action are translated as artifact construction and design argumentation respectively. It is maintained that the two are mutually incompatible modes of working, and that reflection-in-action is only possible if the argumentation is made available within the 'action present' which Schön identifies as that period of time during which reflection can modify action without disruption. This latter requirement is met by being able to access the PHI arguments linked to each critic at the click of a button.

2.1.5 QOC DESIGN SPACE ANALYSIS: Developing and structuring the space around a design

MacLean et al (1991) present a series of objectives for an approach to representing DR which they call Design Space Analysis (DSA). The approach was initially presented in MacLean et al (1989), empirical support for which was subsequently reported in MacLean et al (1990).

The approach uses a semi-formal, argumentation-based notation called QOC (for *Questions, Options* and *Criteria*) to systematically represent and reformulate views of the 'design space' around a design. Questions are used to encapsulate key issues which shape the design, Options are alternative answers to Questions, and Criteria are appealed to in choosing one Option over another. In addition, *Assessments* are the relationships between Options and Criteria (*supports* or *objects-to*), and *Arguments* are used to conduct debate about the status of the above entities and relationships. These elements are summarised in Figure 2.5.

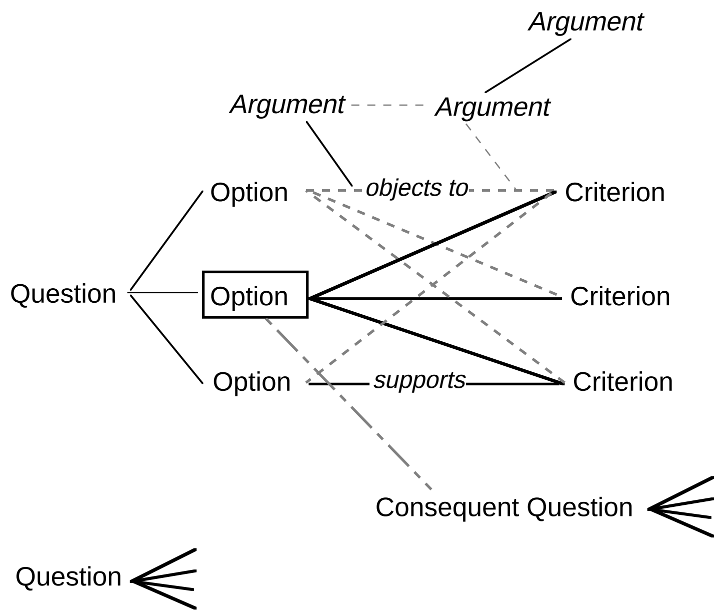


Figure 2.5: The vocabulary of the QOC notation, used to represent Design Space Analyses.

The DSA approach asserts that a DR best serves its purpose – answering *Why?* questions about a design – if it clarifies how the design sits in relation to alternative designs in the domain. For example, text-editors differ from each other along a range of dimensions, some more fundamental in shaping their design than others, e.g. keyboard-based vs. mouse-based; teletype vs. bitmapped display; the way in which directories can be viewed; the distribution of commands between dedicated keys, menus, and key-bindings; the methods of document navigation; the extent to which the editor can automate repetitive operations such as page numbering, bibliographies, and so forth.

The *design space* is this set of conceptual relationships or dimensions, used to compare designs and alternative ways of instantiating those concepts. The design space can never be represented in its entirety, as one can continue to ask Questions to an infinite level of detail, and from numerous perspectives (e.g. implementational vs. human factors). DSA is the process of discovering the important dimensions in a space (what are the key Questions?), of exploring the space of alternatives (Options) which define the local spaces around those dimensions, and of justifying why one point in a local space is better than another (through Criteria and Arguments). QOC also invokes the concept of *bridging Criteria* as a means of focussing decision-making. A bridging Criterion, although not normally represented differently from other Criteria, applies a general Criterion in a particular context, set by the Question. Thus, a high level Criterion such as *simple conceptual model*, might be contextualised for one Question as *familiar concrete metaphor*, in another as *familiarity of icons*, and in another as *consistency with goal structure*.

The results of the DSA are expressed as QOC structures. If QOC is used concurrently whilst designing, the DSA is tracked, and possibly driven by the QOC structures which are constructed (the extent to which QOC aids DSA as opposed to passively reflecting it, is one of the key issues in this thesis). Making the design space explicit in this way, especially if represented in some form of hypertext, allows the designer to restructure the space (by reformulating Questions), as new insights arise into how the design can be best viewed. This emphasis on developing a *logically coherent representation* is perhaps the defining characteristic of QOC DR. The actual design activity *explores* the space—the designer *reflects* on the structure of the space as represented in QOC, and *rationalises* it to clarify the key issues, both for the ongoing design process, and to communicate reasoning to others (Parnas and Clements, 1985). It can be seen that a DSA is not simply a record of the design process which shows *how* the design space was explored, with all the dead-ends, and misconceptions which are inherent to design problem solving; the latter approach characterises an approach to DR which emphasises much more strongly the capture of design *process* [§2.1.1].

2.1.6 DECISION REPRESENTATION LANGUAGE (DRL): Exploring the potential for computation over design rationale

Lee's work on a tool (SIBYL) for managing decision rationale has two main features of interest. The first is the work involved in developing the language on which SIBYL runs, called Decision Representation Language (DRL). Lee and Lai (1991b)[2] present a systematic analysis of the DR domain in order to compare different DR notations in expressiveness (their coverage of the domain). They then demonstrate how the DRL vocabulary is capable of making aspects of design deliberation explicit which can only be handled incongruously by notations with more restricted vocabularies.[3] The DRL vocabulary and an example are shown in Figure 2.6. Note that a user of SIBYL does not interact with graphs in their entirety such as those shown below; SIBYL displays filtered versions (e.g. Claims for a specific Goal and Alternative), and computes evaluation matrices for assessing Alternatives to a particular problem.

2 An extended version of this paper can be found in Lee and Lai (1991a).
3 This property is captured by the cognitive dimension of role expressiveness, discussed later [§10.6].

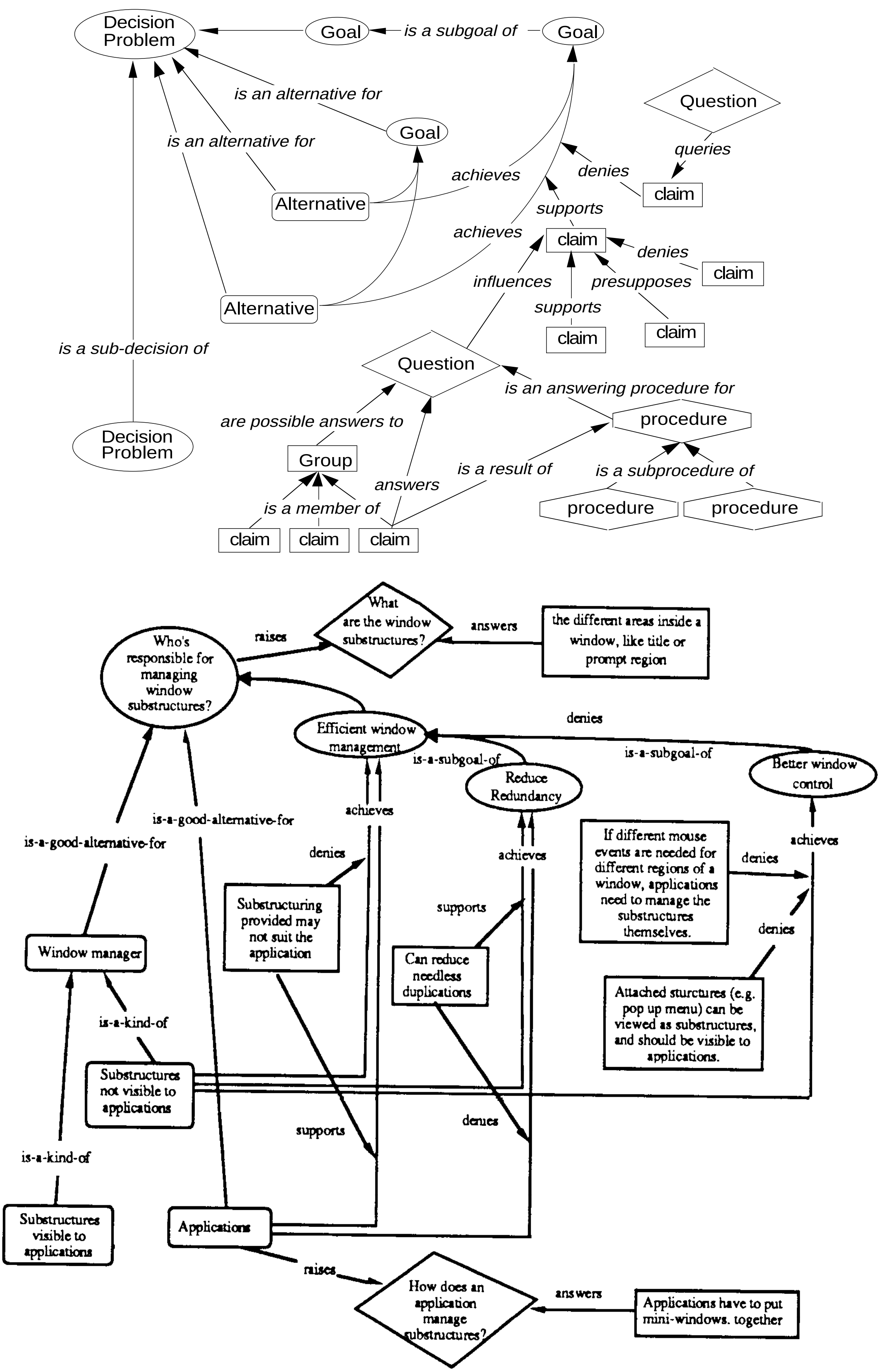


Figure 2.6: The DRL vocabulary, and an example (from Lee, 1991).

The second feature of interest is the knowledge-based tool itself, SIBYL. Lee (1990) argues that the most important payoff which creating DR offers is the possibility of computationally manipulating the DR to explore the implications of different avenues of design before making a commitment. He describes how DRL can be used to provide dependency management (monitoring decisions which depend on each other), precedent management (other decisions sharing same Goals), viewpoint management (arguments sharing common assumptions), and plausibility management (the strength of supporting argumentation for an alternative).

DRL structures are meant to be constructed asynchronously, rather than during real time discussion. Lee (1991) points out that this allows more time to choose between the more subtle entity/relation types which one might wish to use for computational purposes.

Lee (1991) also describes how DRL's greater expressiveness can be coupled with the representation of design artifacts to extend the model proposed by Potts and Bruns (1988 – described above).

2.1.7 Extracting design rationale using claims-analysis: Making the usability consequences of design decisions explicit

Carroll and Rosson (1991) present an approach to constructing *psychological design rationale* by analysing designs. Their approach to integrating HCI theory in evolutionary design which has developed over recent years (Carroll and Campbell, 1989; Carroll and Kellogg, 1989; Carroll et al, 1990, 1991), asserts that artifacts (in their case, user interfaces and documentation) embody theories of usability; that these artifacts can be analysed in a systematic way to make explicit these implicit 'claims' about user psychology; and that these claims can then be abstracted to build a 'contextualised science' – a systematic body of knowledge grounded in actual designs, and which supports subsequent design by guiding the use of usability scenarios to address relevant user issues. The sets of claims which the design embodies constitute what Carroll et al refer to as *psychological DR*. This reciprocal relationship between systems in use and the abstract knowledge about their design is summarised as an information flow around a task-artifact cycle.

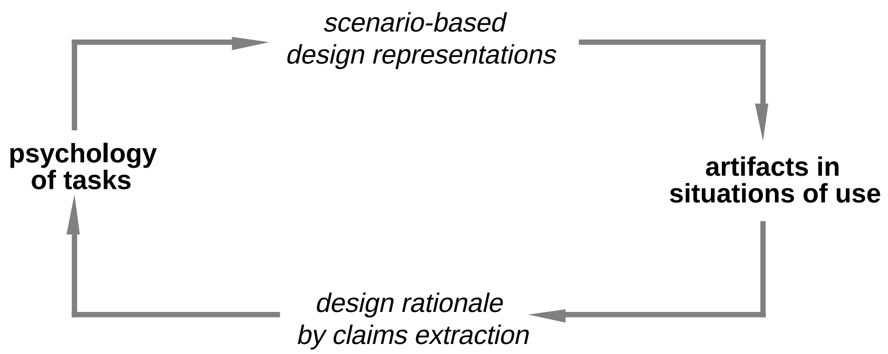


Figure 2.7: The role of DR in the task artifact cycle (from Carroll et al, 1991).

DR by claims extraction is the strategy being adopted towards building an "action science of HCI" (Carroll and Rosson, 1991). A brief extract from a claims analysis is reproduced in Table 2.1 by way of example. Note that these are not necessarily the beliefs or intentions of the designer – simply claims about usability made by the final design.

What can I do?
Exploring demos helps new users learn by doing *(but scripted demos offer little for the user to do)* *(but the demos may not be paradigmatic applications)* *(but learners may have difficulty in finding the corresponding code)*
Browsing and editing in the class hierarchy browser establishes core Smalltalk programming skills *(but may reduce time spent on instantiating and analysing objects)*
How does it work?
Analysing an application with multiple system tools supports convergent reasoning *(but integrating information across tools may be difficult)*
Inspecting an object's instance variables across time supports mapping between its state and behaviour *(but learners must first find or create a useful object to inspect)*
How do I do this?
Interesting features of existing applications evoke subgoals *(but a particular subgoal may be difficult or impossible to pursue)*
Navigating the class hierarchy supports unintentional learning *(but searching for specific classes can be frustrating)* *(but the size of the hierarchy may intimidate learners)*

Table 2.1: Extract from a claims analysis of a programming system, from a learning perspective. The claims are the psychological design rationale implicit in the design (adapted from Carroll and Rosson, 1991).

Claims have what are termed *downsides*, which are possible negative trade-offs. The above claims are driven by three generic questions which focus on issues of particular concern in a learning situation, namely, *What can I do? How does this work?* and *How do I do this?* Claims are essentially implicit hypotheses, which once made explicit can be evaluated. The reification of psychological DR as claims is presented as a way of imposing discipline on activities which are evidently already part of much design practice, but also as a way to improve and support the natural process of design evolution.

Carroll and Rosson note that their focus on *user concerns*, *artifact features* and *psychological consequences* is similar to QOC's *Questions*, *Options* and *Criteria*. One might further add that as a *claim* makes a connection between a feature of the design and a psychological or task-related consequence, it serves a similar function to a QOC *Assessment*.

2.1.8 Research related to design rationale

Three research projects are briefly summarised below, which although not directed towards developing ways of capturing or representing DR as such, share common interests, and have the potential to offer DR research valuable insights.

2.1.8.1 Focussing design on concrete problems

Lewis et al (1991) present an autobiographical account of the process followed during a design project. They assert that this process was typical of much design, but was in conflict with the authoring demands of argumentation-based DR notations. Their key argument is that capturing decisions as DR (e.g. in IBIS, QOC or DRL) was unnatural and distracting as it required them to abstract away from the concrete problems, alternatives and reasons with which they were working. Instead, they argue that design is most naturally a cycle of evaluating suggestions against realistic problems – "what are the problems with *this* idea?" – in order to generate new suggestions to overcome the problems. Consequently, *concrete problems* should be the central construct in design, and abstractions are only recorded if they arise naturally in discussion. The path through the design space is represented as transitions from an initial problem, to alternatives, to new problems, and so forth. This view of design is considered in relation to the data once the studies have been reported [see §10.1].

2.1.8.2 Use of Toulmin to justify decisions in safety critical systems

The influence of Toulmin's (1958) model of argument structure on DR research has been noted [§1.3.1.2], as was its use in writing tools [§1.3.2]. Another application of this formalism is briefly noted here.

Safety-critical systems are generally defined as those whose failure could cause loss of life or damage to the environment. Work is in progress on a current project to explore the use of explicit argumentation to make explicit the reasons for believing that the risks associated with decisions are acceptable (McDermid, 1991). The Safety Argument Manager (SAM) is a prototype tool for safety argumentation, based around Toulmin structures. Envisaged scenarios of use for a fully developed tool include:

- post hoc reconstruction of safety arguments (to better understand the mechanisms used, or possibly for accident investigation);
- clarification of arguments for public debate over the installation of safety critical systems;
- support for high level decision making during software design: consequences of decisions could be evaluated, and decisions recorded for later reference;
- support for system implementation: the tool could integrate formal system models, theorem proofs, and other detailed implementational data from system tests into the Toulmin formalism (e.g. as backing argumentation).

Clearly, this work overlaps in significant areas with DR concerns through its interest in augmenting, rationalising, and communicating arguments. It is suspected that safety argumentation and DR research could learn from each other as the two fields mature.

2.1.8.3 Developing an Argument Representation Language

The EUCLID project (Smolensky et al, 1988) is engaged in developing computer-based support for constructing, communicating, and evaluating different kinds of theoretical argumentation. Smolensky et al provide a useful overview of the issues involved in developing argumentation support tools, setting EUCLID's work in context. EUCLID arguments can be represented in graphical (node-link) or matrix format. EUCLID is based on a predicate calculus representation, called Argument Representation Language (ARL). Whilst this is a much richer language than DRL for instance, EUCLID is still semi-formal—only the structure is formalised, leaving the semantics of arguments informal natural language.[4]

It is proposed that ARL could be used to characterise and constrast different syles of argumentation, such as those found in academic research as opposed to law. It is suggested that one might even differentiate styles found in different fields of academic research, e.g. computer science vs. philosophy vs. linguistics. At present, the tool has undergone several prototype iterations, with some exploratory study into its usability and potential benefits (Hair and Lewis, 1990), as described earlier [§1.3.3].

2.1.9 Reports of DR in use

The key features of several approaches to representing DR have been presented. Before they are discussed, attention turns to review the limited knowledge which we currently have of how designers have used, or failed to use, some of the above notations/tools. As the process of DR use is of prime interest in this thesis, at this stage, the results from the different studies are simply reported. Discussion of their implications is deferred until the QOC studies have been presented [§10.1].

Yakemovic and Conklin (1990) report the only large scale, real world study of DR in use. Over a two year period, a hierarchical indented text IBIS notation (itIBIS) was used by a small development team, using conventional text-editors. The following observations were made:

- given hands-on training, itIBIS was learnt quite quickly;
- use of itIBIS led to more complete and consistent meetings records (notetaking role rotated round team members);
- there was improved analysis of problems, and during conversion from itIBIS→gIBIS, detection of errors in decisions led to cost effective improvements;
- more effective meeting structure, and more precise communication within organisation (*if* itIBIS was mutually acceptable as a formalism);
- quicker reference to previous decisions;

4 Note that efforts have been made by others to formalise the content of arguments, for instance, The OWL system, which uses propositional calculus (McGee, 1986).

- potential support for project management via gIBIS(e.g. tracking unresolved issues; action items raised by decisions).

It was concluded that using a form of IBIS could work in three ways (each of which assumes IBIS training and acceptance on the part of the design team): (i) in qualitative decision support; (ii) as a minimal way of structuring conversation; and (iii) as a means of managing group memory. Yakemovic and Conklin argue that the benefits observed are inherent to IBIS, and that the same results would not be obtained by simply structuring conversation without the IBIS classes—they assert that the extra overhead demanded by using this rhetorical model directs attention towards relevant issues.

As noted earlier, work has been addressing the development of a real-time collaborative version of gIBIS, called rIBIS. Rein and Ellis (1991) briefly report on users' experiences with the tool, taken from 16 meetings. All but one of these meetings was described as "mostly unsatisfying and frustrating" by their participants, with significant difficulties being encountered in using the IBIS method to structure discussions. It was concluded that the main causes of the problems were participants' inexperience with IBIS, and the complexity of the rIBIS user interface. With experience in the method and tool, work progressed more effectively.

Treu et al (1990) briefly report that reasons for decisions were recorded in the design of a modelling and simulation application, and express the view that user interface designers should be accountable to users by making rationale available (in a similar way to that proposed by MacLean et al, 1989). They argue that the effort involved should pay off, but do not discuss how designers might be motivated simply to inform users of their reasoning. Treu (personal communication) expressed the following view, based on the project's experiences in recording DR:

> It at least caused us (the design team) repeatedly to revisit and reflect on decisions we had reached, comparing them deliberately with available options (that might otherwise have been missed) and thereby leading us to formulate justifications for what we did. This questioning process should tend to stimulate designer thoroughness. It can also serve the purpose of consciously developing and encouraging patterns in decision-making, such as repetitions of decisions under similar circumstances. This should tend to promote consistency in design.

McCall's studies of design students using PHI (reported in Fischer et al, 1991) showed that it was extremely difficult to use during the actual 'construction' phases of design (i.e. development of the solution, as opposed to preparatory design discussion). This was despite the fact that the problem clearly *was* being decomposed into subissues by the designers, in accordance with the PHI method. Designers found themselves being sidetracked into discussions about philosophical design issues, extended analysis and requirements, and other abstract issues, rather than "getting into the design." These results lend support to Lewis et al's problem-centred approach, in so far as the latter was proposed in reaction to the difficulties experienced in formulating more abstract descriptions of ideas.

Fischer et al interpret these results as support for Schön's theory [§1.4, §2.1.4]. According to this account, use of PHI on its own prolongs the reflective process, blocking the 'real design activity' characterised by the use of construction expertise (knowing-in-action). There are strong predictions which one could make about the way in which a tool based on Schön's ideas would be used. However, and perhaps crucially, evaluations of JANUS have focussed primarily on designers' reactions to the critics (Fischer et al, 1988; 1991), rather than the way in which the designers used the DR in relation to their design work. Data needs to be collected on whether designers are in fact able to use JANUS/PHIDIAS argumentation effectively, and how the concepts of knowing- and reflection-in-action relate to that data.

2.2 Comparing and contrasting the approaches

This discussion section identifies commonalities and tensions between the approaches, and highlights the most important open research issues facing the field.

2.2.1 Summary of properties of the different DR approaches

As will by now be clear, the different approaches to DR make overlapping, as well as differing claims regarding the roles which DR could play. In order to clarify the various emphases, the approaches have been compared against four dimensions to provide an overview of the similarities and differences between them (Table 2.2). The dimensions were defined as follows:

- *reusability*: How generic is the DR produced? To what extent can the design/HCI knowledge embodied be retrieved and applied in other design contexts?
- *design process capture*: How much of the process involved in developing the design is reflected in the DR? What narrative elements are available, i.e. the original form and order of ideas?
- *computational services*: What kind of operations does the DR allow the system to perform? This strongly determines the ease of DR management (maintaining a consistent DR), the number of different structures which can be reified (goals; viewpoints; dependencies), and DR retrieval and reuse.
- *authoring overheads*: What are the overheads, for a trained designer, in constructing DR within a particular approach?

As far as possible, ratings (ordered none–poor–ok–good–v. good) are based on the characteristics of the *approach*, rather than on how developed a particular instantiation of it is (e.g. how advanced a particular tool is). For instance, any of the notations (except perhaps Claims analyses, not being semi-formal) would benefit from the services offered by SIBYL, the structural queries offered by PHIDIAS, or the syntactic checking offered by Potts and Bruns' Prolog knowledge base (indeed, as noted earlier, the Potts and Bruns approach has been analysed by Lee in terms of DRL). However, some approaches receive lower ratings on this dimension since they place an emphasis on real-time use during deliberation which constrains the complexity of the DRs which can be produced.

Ratings in italics indicate that the approach's developers explicitly claim to support a particular function; other ratings result from personal analysis of the approach. The rating is briefly explained beneath the respective cell. The reasoning behind the ratings is open to debate (most likely by the approaches' developers); one difficulty is that most of the approaches are relatively new, and it is not clear, or made explicit in their descriptions, how they relate to each of the dimensions. For instance, there has been little description in Carroll et al's view of psychological DR, of how the DR might be managed. Nor is there any evidence as to how easily gIBIS structures can be reused, or JANUS PHI structures authored. However, one reason behind such an exercise as this is precisely to highlight those areas on which each approach has majored, and those where, thus far, little work has been done.

Approach to DR	**Properties of approach**			
	reusability	***design process capture***	***computational services***	***authoring overheads***
Design process capture using gIBIS	**poor**	***v. good***	***ok***	**ok**
reason	*DR captures process of one design only, with little abstraction or reorganisation of issues*	*DR constructed during meetings, with minimal reconstruction*	*small vocabulary for simplicity*	structuring discussion as IBIS rhetorical moves is not natural
Tailoring DR to software engineering methods	**good**	***good***	***v. good***	**poor/ok**
reason	more rationalis-ation than in gIBIS	*most types of argument can be represented*	*representation of Artifacts supports integration with CASE tools*	larger more exp-ressive notation may take longer to construct
Integrating argumentation with construction	***good/v. good***	**none**	***good***	**poor**
reason	*PHI issue bases constructed to be modularised (but critics may constrain range of design domains)*	PHI constructed beforehand, not during design	*critics maintain consistency; PHI*	construction of reusable PHI structures, and linking to critics requires effort
QOC Design Space Analysis	***v. good***	***none/poor***	***ok***	**poor/ok**
reason	*a DSA relates multiple designs in a comparable form*	*narrative elements filtered out in rationalisation*	*small vocabulary for simplicity*	restructuring design space requires effort
Computing over DR with SIBYL	**good**	***poor/ok***	***v. good***	**poor/ok**
reason	consequences of changing variables can be assessed; useful precedent management	*most types of argument can be represented, but expressive lang-uage too slow to use in real time*	*several powerful services supp-orted for manag-ing, linking and retrieving DR*	larger, more expressive notation may take longer to construct
Extracting DR by claims analysis	***v. good***	**none**	**ok**	**poor**
reason	*abstracted claims build an action-science grounded in design*	claims extracted from existing designs; irrelevant to actual process	not clear how claims can be checked, linked, or viewed	much craft expert-ise needed to extract useful claims

Table 2.2: A comparison of approaches to DR against a range of properties. The key feature of each approach is highlighted. Claims made explicitly by the developers of an approach are in italics.

2.2.2 BALANCING USABILITY, COMPUTABILITY, AND EXPRESSIVENESS

The trade-off between optimising systems in favour of the user or computer is familiar in user interface design generally, and is equally important in designing argumentation models which will be both usable and computationally powerful. The trade-off between computational versus human tractability is an issue which several DR researchers have identified as critical (Newman and Marshall, 1990; Conklin and Yakemovic, 1991; Lee, 1991). The key trade-off is in how refined the entity and relationship classes are. Whilst expressiveness increases as more unique elements are added to the vocabulary, clearly there are usability implications which dictate that no more distinctions between node or link types need be made. Designers are first and foremost *designers*, not experts in dialectics. The designer must not be 'overpowered' by the range of types offered, or fields to fill in when, for instance, he wishes simply to add a new Criterion. The usefulness of a large vocabulary is limited by the user's knowledge and understanding of the domain, and of argumentation. Are the different types useful, or even meaningful? Malone et al (1988) have demonstrated with Object Lens that when implemented well, users can, and do refine object classes when they feel that there is sufficient payoff. This issue is also discussed in terms of cognitive dimensions of DR notations [§10.6].

The issue of interest in the studies reported in this thesis is the extent to which designers are able to learn to use the relatively simple QOC notation. Taking into account other factors such as user expertise, and the quality of the user interface to any future DR environments, the fact remains that if designers encounter difficulties with QOC, there may be significant implications for developers of more elaborate representations. In order to map out the 'usability' space of DR formalisms, more empirical studies are needed which explore the dimensions of *DR notation and tool*, *problem structure*, and *user expertise*.

2.2.3 SUMMARY: KEY ISSUES FACING RESEARCH INTO DESIGN RATIONALE

From the above review and discussion, several key issues can be seen to create tensions across the 'design space' of DR approaches and notations:

- ***How much to modify existing design practice?*** To what extent should we try to modify designers' problem solving in our attempts to improve it? Should we passively record discussion, or shape it in some way? When to record it?
- ***Rationalised versus narrative DR?*** Is DR a knowledge-representation of the design space for relating multiple designs, or a narrative of the design process for one design?
- ***Reusability versus design specificity?*** Closely dependent on the last issue, in a more generic, coherently organised DR, the reasoning can be treated as a reusable, semi-formal reference point for design expertise; it takes much more work to abstract the reasoning from a process-oriented DR for re-application in other design contexts.

The extent to which the DR can be manipulated computationally is a factor in each of the above, and is consequently a recurring theme thoughout the thesis discussion.

Chapter 3

Study 1: Cognitive Issues in Retrieving Design Rationale

3.1 Introduction and experimental methodology

3.1.1 AIMS OF STUDY 1

This first study explores cognitive issues in retrieving design rationale. As DR must have software support if it is to be viable, the relationship between DR and potential tool functionality is critical. In tracing the research roots of DR, it was observed how semi-formal argumentation schemas, represented graphically as entity-relationship structures, have come to be a major representation for DR. However, despite considerable research activity in building hypertext tools for managing structure, very little work has addressed the cognitive issues in using graphical browsers, the primary interaction medium of such tools. Part of this study, therefore, includes an initial exploration of how users might behave when retrieving DR on a hypertext system which offers substantial browser functionality. This was achieved in this and the next study by simulating a DR database tool, offering multiple representations of the DR in different formats and levels of detail, plus an additional representation called a Criterion tree which shows inter-Criterion relationships. The aim was to identify properties of QOC representations which when supported computationally, will facilitate the cognitive task of locating appropriate DR.

The three main questions which this study explores are:

- what are the cognitive tasks involved in retrieving relevant DR about a design?
- what developments to the representations used in this study – QOC graphs and Criterion trees – are necessary for more adequate DR tool support?
- what are the cognitive overheads of working with multiple DR representations varying in physical structure, level of detail, and viewpoint?

Within the small DR research literature currently, there is relatively little reference to the sorts of queries QOC should support, or the cognitive aspects to this problem. However, before the details of the study are described, it is helpful to review the implications of existing research efforts into DR retrieval issues.

3.1.2 EXISTING RESEARCH ADDRESSING DR RETRIEVAL

In the introduction to their analysis of the domain of DR, Lee and Lai (1991b) propose that a DR representation should support a set of representative questions. These are reproduced below, added to which are notes on the functionality (i.e. the sorts of entities and attributes) which a DR system (i.e. a notation plus environment[1]) would need:

Question	**Notation needs to represent:**
What is the status of the current design?	e.g. goals satisfied unresolved issues
What did we discuss last week and what do we need to do today?	nodes with a *time* attribute plus diary facility to flag decisions to be made before deadlines
What are the alternative designs and their pros and cons?	alternatives, criteria and assessments
What are the two most favourable alternatives so far?	some measure of the extent to which alternatives satisfy criteria
Sun Microsystems just released their X/NeWS server – how does the release change our evaluations?	a metric for comparing alternatives
What if we consider portability?	some way to monitor relationships between criteria, and a measure of most favoured alternative
Why is portability important anyway?	meta-argumentation about the validity of criteria
What are the issues that depend on this issue?	inter-issue dependency relationships
What are the unresolved issues? What are we currently doing about them?	issue attribute of *resolution status*
What are the consequences of doing away with this part?	relationships between decisions, e.g. consistency, dependencies
How did other people deal with this problem? What can we learn from past design cases?	search facility to locate argumentation using structural and content descriptors

Table 3.1: A set of realistic queries which a DR system should be able to answer (as proposed by Lee and Lai, 1991b), annotated to show the entities and attributes needed to provide the necessary information.

Work on using QOC as a representation for communicating human factors knowledge has been described by Carey et al (1991), based on McKerlie (1991). They have explored the potential of a tool which allows designers to access a database of Questions, Alternatives (illustrated by Scenarios), and Criteria, in order to locate examples of human

[1] This definition of *system* derives from Green (1989); see cognitive dimensions analysis of DR notations [§10.6].

factors expertise appropriate for a given problem. Whilst the results and validity of this approach are not of direct relevance in the present context, it is of interest to note the questions which McKerlie (1991) anticipated in building the tool (p.83):

- inquiries about specific design questions phrased in functional terms (e.g. *how should users cancel operations?*)
- inquiries centred on specific features or objects (e.g. *what are the possible uses for the Escape key?*)
- inquiries based on general criteria (e.g. *ease of learning is an important criterion for my project—what are some of the implications to be considered?*)
- inquiries based around a specific existing system (e.g. *the XYZ system was targetted for similar users with a different task—how were windows designed there, and why?*).

Despite the fact that McKerlie et al's tool was not intended to be used in quite the same way as other DR systems, the above could be candidate queries which any DR system should answer. They illustrate the typical perspectives and concerns which one can reasonably expect users of future tools to bring with them.

The work of Fischer et al (1991) on JANUS represents a novel approach to the DR retrieval problem (described in [§2.1.4]. The PHI structures which JANUS uses are in fact similar in nature to QOC design space analyses (in terms of the rationalisation and issue abstraction invested in their construction). The retrieval problem is solved by linking what is judged to be relevant argumentation to messages about CAD design; rules fire when design guidelines are violated in a CAD system. Fischer et al also argue for the importance of *reusable* issue bases, to help reduce the effort of DR construction at the start of each new project. The idea is that particularly in routine design, DRs could be constructed in part by 'bolting together' different DR modules which import recognised, validated arguments to support decisions in well understood areas of the new design. The benefits for DR retrieval are significant: designers would become familiar with the organisation of different issue bases within a common domain, because they would be based on the same core generic structure.

All of the above examples of DR queries represent typical retrieval tasks whose cognitive and representational demands are the focus of this study. With Study 1 now set in the context of existing research in DR retrieval, let us turn to the details of the procedure.

3.1.3 METHOD

3.1.3.1 Introduction to QOC

For the purposes of Studies 1 and 2, the concept of DR was presented as "a record of the decisions which were taken" in the development of a system. Subjects were introduced to

QOC via a brief summary of the function of Questions, Options, Criteria and positive and negative Assessments, and several examples.

3.1.3.2 Experimental task scenario

Subjects were asked to play the role of a system designer about to develop new facilities for the NoteCards hypertext system. The scenario stated that as background research, they wished to explore the DRs for NoteCards, and for one of NoteCard's competitors, HyperCard. This would give a feel for NoteCards, and for the conceptual differences with HyperCard. They have a number of queries about the two systems, for which they are going to use a database system which can present the DR in the form of browsers of different types and levels of detail, any of which are available on request.

3.1.3.3 Subjects

Subjects varied in research experience from research assistants/associates to lecturers. All were engaged in full-time HCI research, as summarised in Table 3.2. The subject numbers shown are used throughout the report.

Subject	Status	Background	Years in HCI	Research interests	Relevant knowledge (HT=hypertext)
1	research associate	psychology	1	structured communication in conferencing	HT: brief use of HyperCard DR: none
2	lecturer	psychology	12	hypermedia; Computer-Aided Learning; user modelling	HT: very familiar with concepts and some systems DR: familiar with QOC and applications (but no authoring experience)
3	research assistant	computer science	1	structured communication in conferencing	HT: not familiar DR: not familiar
4	research fellow	computer science	2	formal modelling/ system design methods/ logic programming	HT: familiar with general concepts & HyperCard DR: familiar with general concepts
5	research associate	psychology	2	interface evaluation/ knowledge elicitation/ user & system modelling	HT: familiar with general concepts and HyperCard DR: familiar with general concepts
6	research assistant	computer science	1	formal methods/ executable specific-ations/ prototyping	HT: familiar with general concepts and HyperCard DR: familiar with general concepts
7	lecturer	psychology	6	usability evaluation/ theory in HCI	HT: very familiar with general concepts and some systems DR: familiar with general concepts

Table 3.2: Subjects used in Study 1 (and also in Study 2)

All but one subject was familiar with the general node-link model underlying hypertext, some had used HyperCard, but none were familiar with the NoteCards system in any detail. Similarly, most had received some exposure to the basic concept of representing design rationale explicitly, largely through a particular research paper (MacLean et al, 1989) or presentation. The exception was S2, who was involved on a project in which QOC was being used (by another party), and was as such considerably more familiar with its concepts, although not in using it himself.

3.1.3.4 DR representations used

There were three types of QOC browser available to subjects: QOC graph structures, indented text QOC, and Criterion trees, also available as graphs and indented text.

3.1.3.4.1 Graphs

The graphs represented QOC in the familiar node-link format, an example of which is shown in Figure 3.1. Decisions were shown in bold to highlight them, as were Questions. Criteria which supported Decisions were also bold, as explained in the description of Criterion trees.

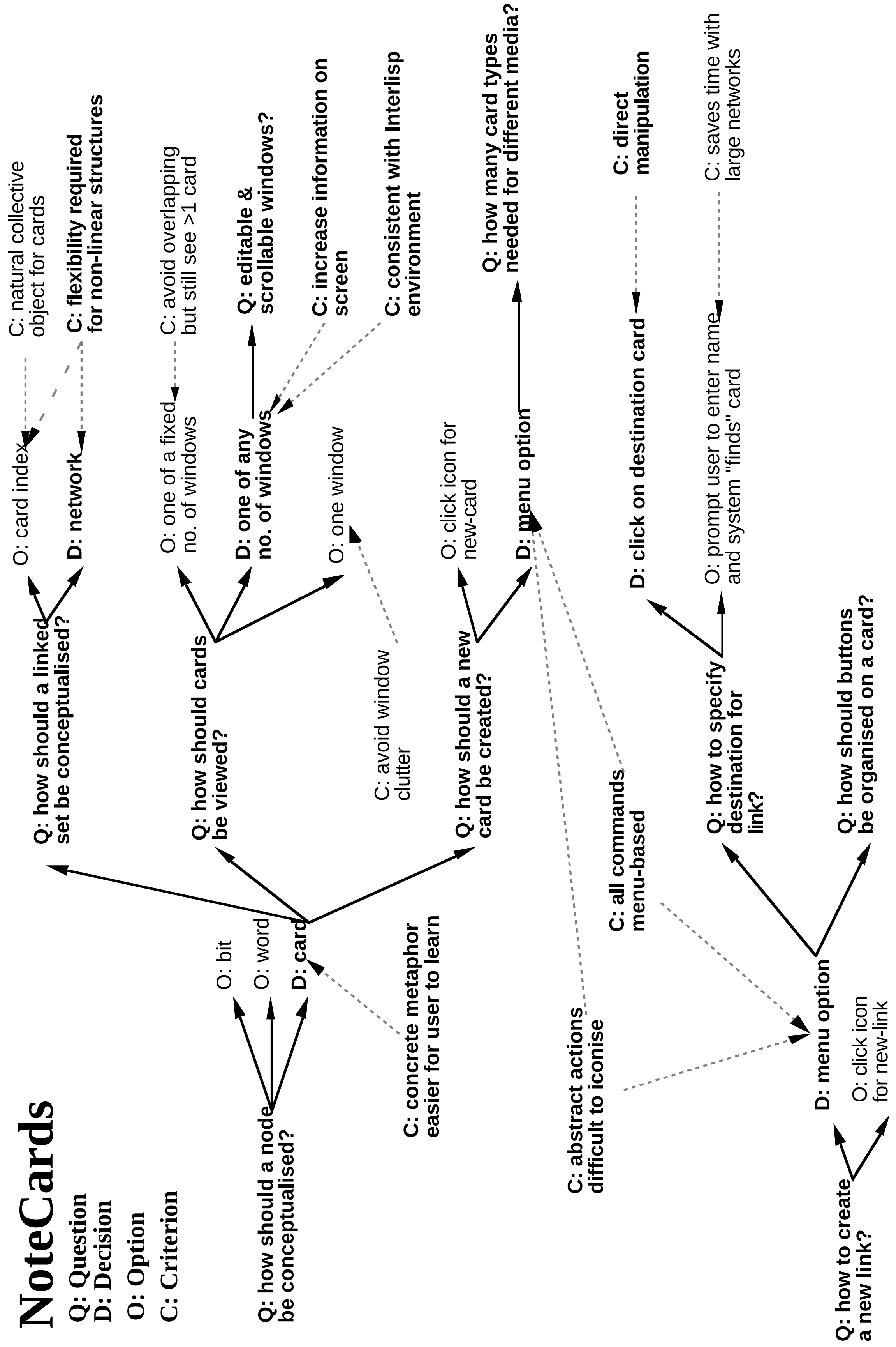


Figure 3.1: Part of a QOC graph (unfiltered).

3.1.3.4.2 Lists

In order to see if there were any circumstances under which subjects preferred textual to graphical representations of QOC, linear, hierarchical lists were created using indentation to indicate consequent Questions, typical of the display in outlining tools. An equivalent example to the above graph is shown in Figure 3.2. Coincidentally, these textual QOC lists are very similar to the indented-text IBIS whose use is reported by Yakemovic and Conklin (1990).

NoteCards List

Q: Question
D: Decision
O: Option
+C: supporting Criterion
–C: objecting Criterion

Q: how should a node be conceptualised?
O: bit
O: word
D: card
+C: concrete metaphor easier for user to learn

> **Q: how should a linked set be conceptualised?**
> O: card index
> +C: natural collective object for cards
> –C: flexibility required for non-linear structures
> **D: network**
> **+C: flexibility required for non-linear structures**
> **Q: how should cards be viewed?**
> O: one of a fixed no. of windows
> +C: avoid overlapping but still see >1 node
> **D: one of any no. of windows**
> **+C: increase information on screen**
> **+C: consistent with Interlisp environment**
> O: one window
> +C: avoid window clutter
>
> > **Q: editable & scrollable windows?**
> > O: no
> > O: text only
> > +C: lower memory load
> > –C: user expectations of Interlisp windows
> > **D: all contents**
> > –C: lower memory load
> > **+C: user expectations of Interlisp functionality**

Q: how should a new card be created?
. . . . . .

Figure 3.2: Part of an indented-list QOC.

3.1.3.4.3 Criterion trees

Whilst lists were an alternative representation of the material in the graphs, the content of Criterion trees was different. A Criterion tree shows the relationships between *general Criteria* which may attain the status of 'design principles,' and more specific *bridging-Criteria,* which relate general Criteria to a particular context [§2.1.5]. All of the bridging Criteria used in a QOC are represented in the Criterion tree, classified under their more general Criteria. A Criterion tree offers a second view of the QOC, making the relationships between Criteria explicit, which cannot be done on a conventional QOC graph.

Criterion trees were devised to meet several experimental requirements:

- subjects should use a different representation to the normal QOC graphs, so that issues associated with *integrating information between different representations* could be studied;
- this new representation should provide a *different view* of the content of a QOC, making different information explicit;
- this representation should be potentially *useful to the intended domain of use* (QOC retrieval); the queries subjects had to answer (see below) were devised as representative information requirements for an outsider wishing to access an unfamiliar QOC – it was suspected that on its own, the basic QOC graph would be insufficient to meet these needs as it presents only one perspective.

Criterion trees were available as vertical tree structures, and as indented hierarchical lists. Figure 3.3 shows a fragment of the NoteCards Criterion tree, which includes Criteria used in the QOC graph and list examples of Figures 3.1 and 3.2.

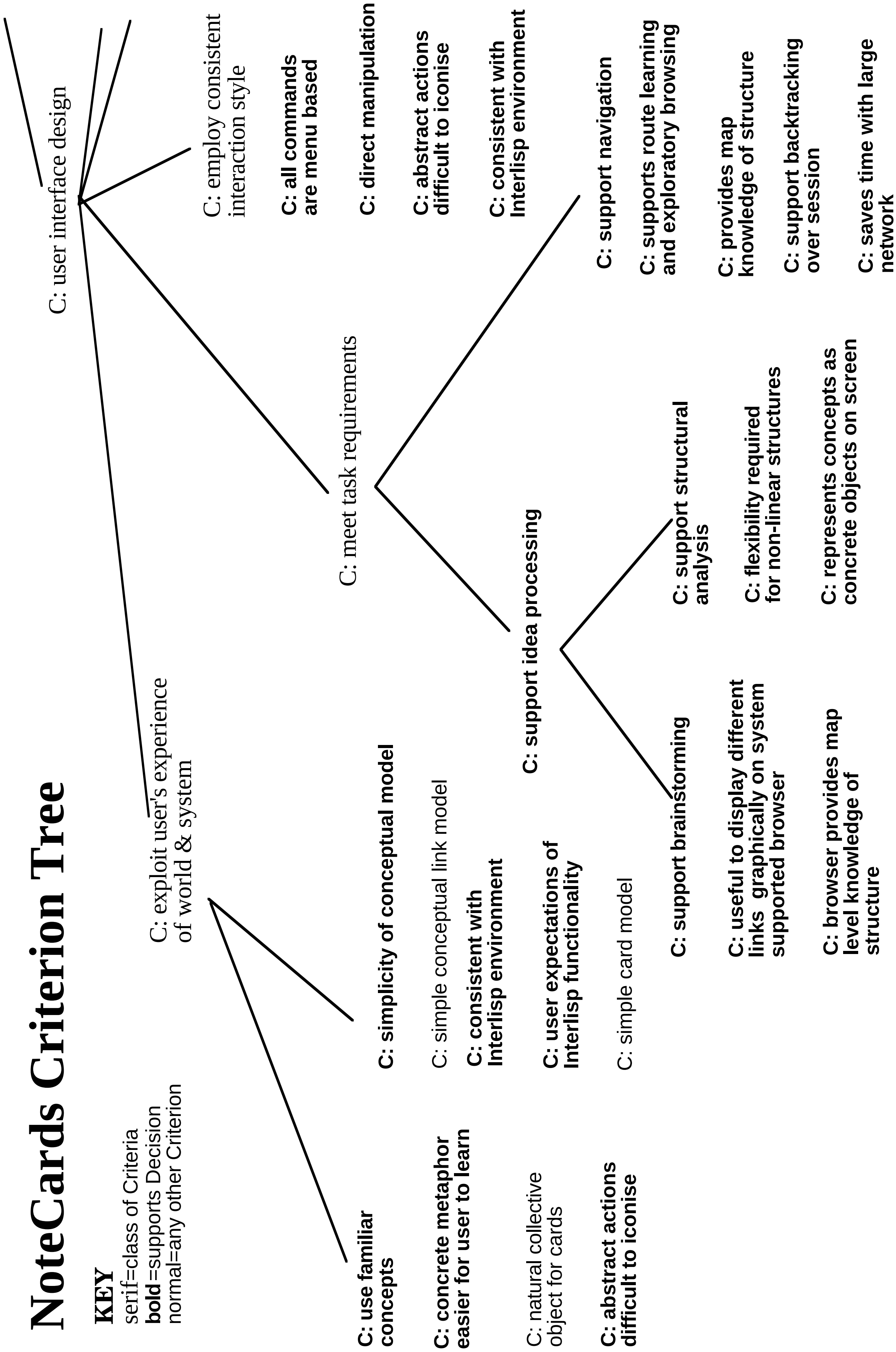


Figure 3.3: Part of a graphical Criterion tree

The last point to note about Criterion trees is the use of different typefaces. This was a simple way to enrich the information content of the trees by providing some indication of a Criterion's status. Clearly, many visual codes could be introduced to highlight useful information (e.g. Criteria which object to Options, or the number of times Criteria are used). To keep the display simple for the subjects, it was decided that only the most important information should be present, namely those Criteria supporting final Decisions. These were displayed in **boldface**.

Thus, a role which Criterion trees could potentially play would be as a means of understanding differences between designs through emphases in the design reasoning, reflected in the extent to which different classes of Criteria supported decisions. Thus, one CAD system might be biased towards high functionality whilst the other towards learnability; one musical composition package oriented to low-end equipment for novices as opposed to high performance platforms for professionals, and so forth. This is discussed towards the end in more detail [§3.3.1].

To summarise, it was envisaged that Criterion trees should support information mapping in two directions:

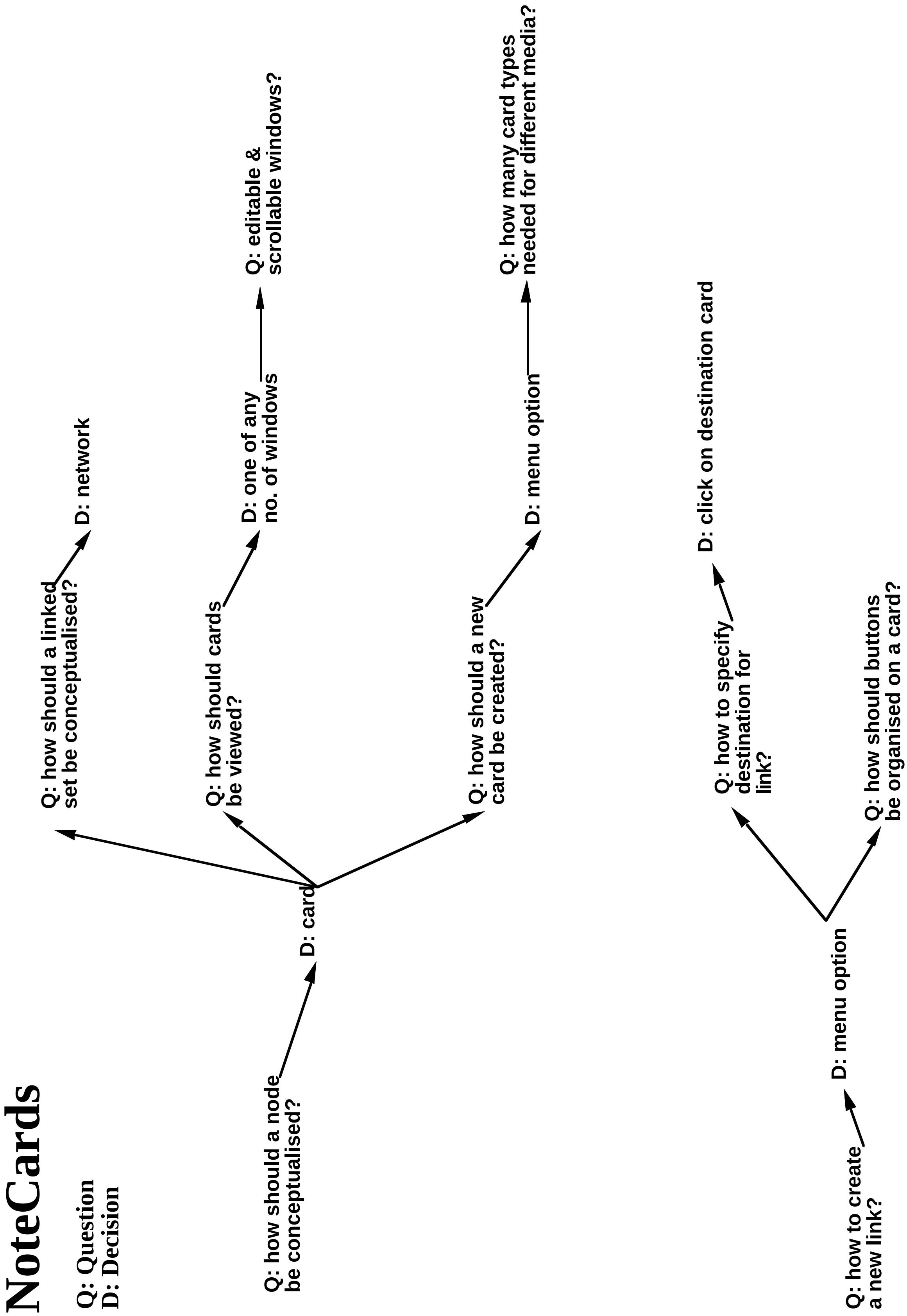
- *from Criterion tree to QOC*: the tree offered a profile of (the most widely used) classes of Criteria, and the extent to which each class supports decisions;[2] it could thus be used for comparing systems and as as index to access QOC.
- *from QOC to Criterion tree*: use of a Criterion tree focussed the Criteria used during design; any Criterion used in the QOC should be justifiable in terms of the more general Criteria which it supports.

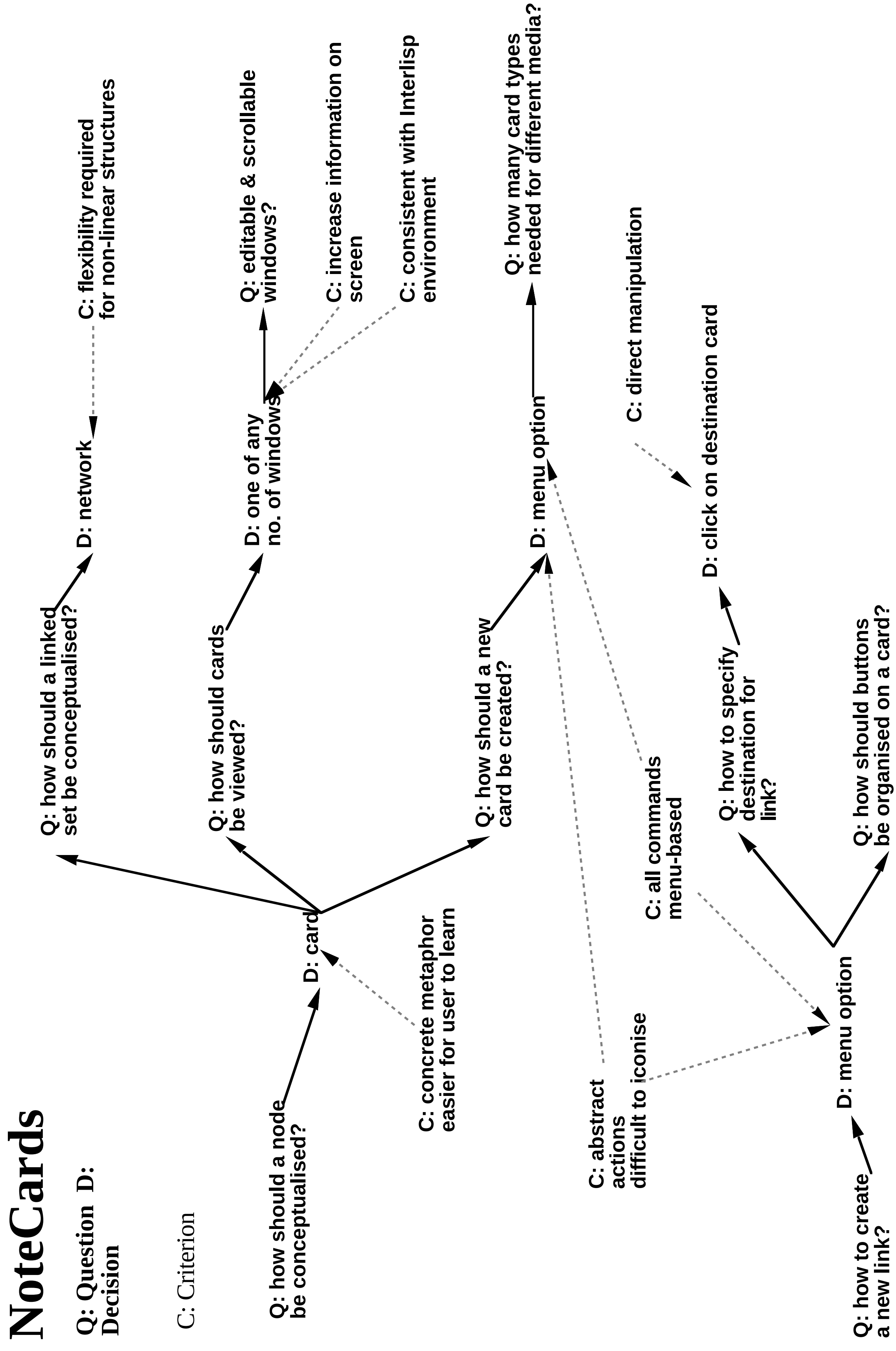
3.1.3.4.4 Filtering the representations

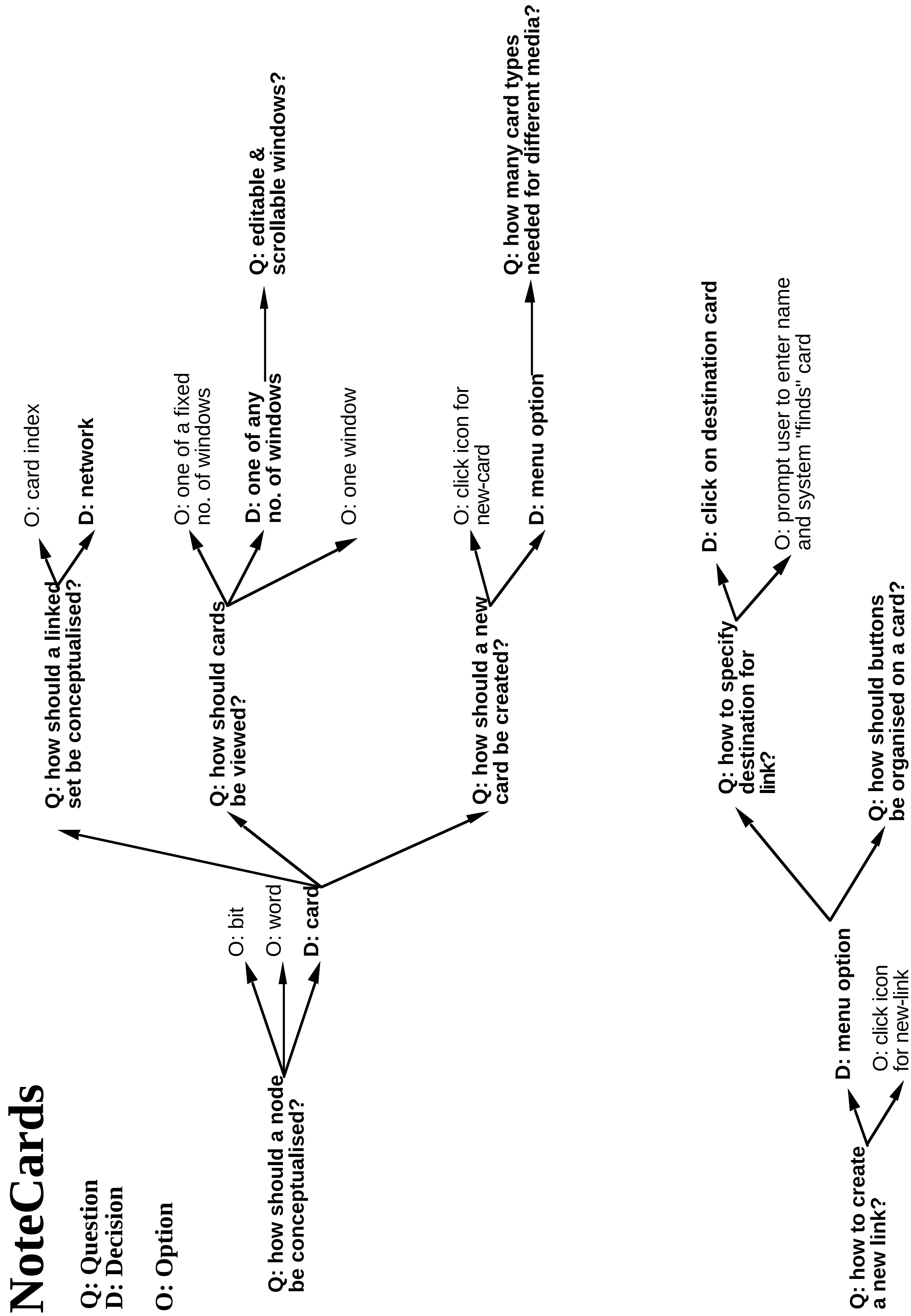
Apart from the perspective offered by a browser, subjects were also able to choose from four levels of detail. The most filtered view showed only **Questions and Decisions**, allowing the user to follow through a sequence of design decisions without excess 'visual noise.' For more detail, the user could add Criteria to this display, to see **Questions, Decisions and Criteria**, or Options, to see **Questions, Options and Decisions**; the most detailed view showed **Questions, Options, Decisions and Criteria**. All of these were available at any time either graphically or as a list. Combined with the Criterion trees (list or graph for NoteCards and HyperCard), subjects had available to them a choice of 20 displays with which to answer the queries. Figures 3.4-3.6 overleaf illustrate the visual effect for different levels of filtering. It was felt that this was adequate to gain useful data about user behaviour when faced with a browser tool for retrieving QOC.

---

2 Henceforth, these Criteria are referred to as 'bold Criteria.'


NoteCards
Q: Question
D: Decision
Q: how should a node be conceptualised?
D: card
Q: how should a linked set be conceptualised?
D: network
Q: how should cards be viewed?
D: one of any no. of windows
Q: editable & scrollable windows?
Q: how should a new card be created?
D: menu option
Q: how many card types needed for different media?
Q: how to create a new link?
D: menu option
Q: how to specify destination for link?
D: click on destination card
Q: how should buttons be organised on a card?


NoteCards
Q: Question D: Decision
C: Criterion
Q: how should a node be conceptualised?
D: card
C: concrete metaphor easier for user to learn
Q: how should a linked set be conceptualised?
D: network
C: flexibility required for non-linear structures
Q: how should cards be viewed?
D: one of any no. of windows
Q: editable & scrollable windows?
C: increase information on screen
C: consistent with Interlisp environment
Q: how should a new card be created?
D: menu option
Q: how many card types needed for different media?
C: all commands menu-based
C: abstract actions difficult to iconise
Q: how to create a new link?
D: menu option
Q: how to specify destination for link?
D: click on destination card
C: direct manipulation
Q: how should buttons be organised on a card?



Figures 3.4-3.6: Three levels of filtering for QOC graphs

3.1.3.4.5 Paper-based representations

The whole experiment was conducted with paper-based representations, due to the lack of a hypertext system which computed graphs, or a large enough screen to display simulated graphs. As a result, the experimenter acted as an imaginary QOC system, presenting new 'browser-windows' on subjects' requests, e.g.

"Can I have the list, for HyperCard, showing Questions, Options, and Decisions"
"Can I have the Criterion tree, as a graph, for NoteCards"

As far as 'physical' interface requirements are concerned, using a paper-based method actually offers a number of advantages over any hardware currently available, indeed near ideal conditions in certain respects: instantaneous 'computation' of browsers, an unlimited number of open windows, a very high resolution display, A2 sized windows, overlapping multiple-windowing facilities, and a desktop about six times the area of a normal screen. All browsers were titled, in the same way that screen-windows have titlebars. From this perspective, problems learning the interface were eliminated, which allowed cognitive issues relating to QOC retrieval to be explored relatively freely.

3.1.3.5 QOC queries to be answered

15 queries were given to subjects, varying in complexity (Table 3.3).

1 - In HyperCard, which kinds of information are scrollable?

2 - How can the user move about in a NoteCards network?

3 - What ways were considered for viewing material in NoteCards?

4 - What were the possibilities in HyperCard for how link buttons could be placed on a card?

5 - Why was NoteCards made a multi-windowing environment?

6 - How does a NoteCards user specify the destination for a new link, and why was this method chosen?

7 - Why does HyperCard not have a graphical browser?

8 - In HyperCard, the user can jump back to a previously visited card — what reasons lie behind the design of this facility?

9 - How do NoteCards and HyperCard compare in terms of navigational methods?

10 - How do the systems compare in the extent to which they take into account basic cognitive psychological characteristics of their users?

11 - Does HyperCard have any advantage over NoteCards in the way in which the destination card for a new link can be defined?

12 - Which of the two systems do you think offers more support to the user in designing buttons?

13 - Is it true that NoteCards gives more control to the user over facilities, than HyperCard?

14 - Which system's design was more constrained by the limitations of its hardware?

15 - NoteCards was developed to support specific sorts of tasks. Identify three decisions in its design which were made specifically with these tasks in mind.

Table 3.3: QOC queries used in Study 1

A table of queries and solution browsers can be found in Appendix 1, which sets out the minimum number of browsers and level of detail which were required to answer each query. Use of more complex browsers than those listed is possible, but less complex browsers are inadequate. Examples of increasingly complex queries are shown below, where Q=Question, D=Decision, O=Option, and C=Criterion.

Query	No. browsers reqd.	Browsers
How can the user move about in a NoteCards network?	1	NoteCards–Q/D
What ways were considered for viewing material in NoteCards?	1	NoteCards–Q/D/O
Why does HyperCard not have a graphical browser?	1	HyperCard–Q/D/O/C
How do NoteCards and HyperCard compare in terms of navigational methods?	2	NoteCards & HyperCard–Q/D
NoteCards was developed to support specific sorts of tasks. Identify three decisions in its design which were made specifically with these tasks in mind.	2	NoteCards–Q/D/C + Criterion tree
Which system's design was more constrained by the limitations of its hardware?	4	NoteCards & HyperCard–Q/D/O/C + Criterion trees

Table 3.4: Some example QOC queries, varying in the complexity of browsers needed to locate the answer.

A query's complexity was defined with respect to the number and/or type of browser required. Naturally, any query could be answered using a more complex series of displays than necessary (which did occur); what is not clear from Table 3.4 is that some queries are impossible to answer without the correct representation. These are ones in which a different view onto the data is involved, specifically, queries requiring a Criterion tree. If, for whatever reason, a subject tried to infer from a normal QOC graph the information represented explicitly in a Criterion tree, severe problems would be encountered, either because of the working memory needed to remember instances of relationships across a QOC (e.g. all Options assessed by hardware-related Criteria), or because it was impossible to deduce Criterion-bridging Criterion relationships from the graph alone.

3.1.3.6 Video and audio data

A camera with directional microphone was attached to the wall above and behind subjects, providing a bird's-eye view of the table and representations which subjects used (Figure 3.7).

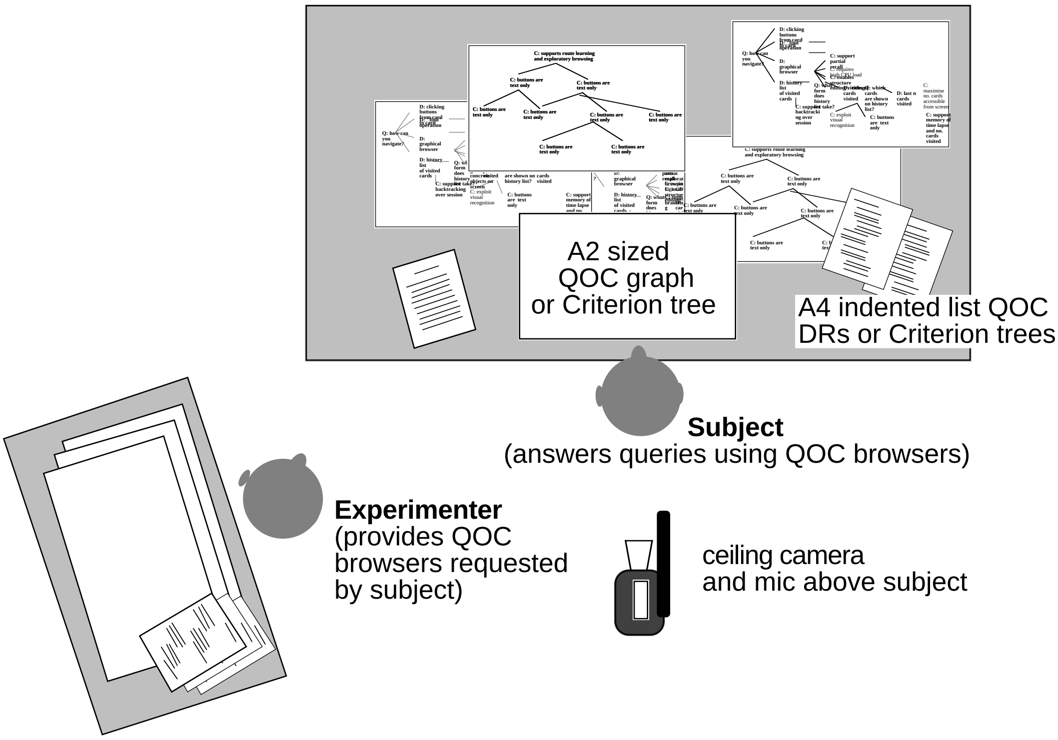


Figure 3.7: Schematic view of the video frame used for analysis in Study 1.

3.1.3.7 Conditions

Although a parametric design of the more traditional form was not required for the purposes of this study, two variables were modified in order to explore potentially interesting differences.

1. S3 was provided only with list displays, whereas all other subjects had the option of using graphs or lists. This was to see how useful the graphical cues were, and the sorts of problems S3 incurred.

2. As a result of observing the first four subjects, the presentation of the queries to be answered was changed for S5-7. Subjects 1-4 were given a list of queries, as in Table 3.3, but it was found to prejudice them towards following the queries in the given order. As the way in which queries are organised was of interest, a more flexible means of presentation was devised. Subjects 5-7 were given a shuffled pile of cards, each one with a query on it, and were told they could tackle them into whatever order they wished. This resulted in substantially different strategies from the earlier subjects. It was felt that this simulated real world conditions more realistically, since users do not normally have queries ordered for them.

3.1.3.8 Procedure

Subjects were allowed to request and close browsers as they wished, and to make notes as needed. They were not told which levels of detail of representation were available; this was to encourage them to request what they regarded as the *ideal* browser for the task without worrying about whether it was actually available. If they requested an unavailable browser, after explaining why they wanted it, they were told it was not available, and the closest alternative suggested. For instance, if a *Decisions & Criteria* browser was requested, *Questions, Decisions & Criteria* was offered.

Subjects were asked to think aloud all the time, commenting on what browsers they were looking at, what information they thought a query was after, whether a browser displayed what they expected, and so forth. The experimenter prompted subjects if they fell silent for long periods, and asked questions if it was not clear what the subjects were doing.

The experimenter ended the experiment after one hour, or as soon as the query currently being dealt with was finished; no subject completed all of the queries. On a few occasions when the hour was about to end, the experimenter gave a specific query to the subject (one of the more complex ones), in order to assess translation into QOC, and use of multiple browsers. These incidents are logged in Appendix 1. Clearly, experimenter selection of queries would be impossible in a controlled, parametric design, but in the context of this exploratory study, it was used simply to maximise the range of data gathered.

3.2 Results and discussion

The results and discussion are organised as follows. First, the process of analysing the data is briefly described. As a means of organising the key results, a task model is then presented which summarises the tasks involved in retrieving QOC in answer to a query, together with the factors which were found to impinge on each task. Lastly, the main results are reported, organised around the structure of the task model.

3.2.1 DATA ANALYSIS PROCESS

The process of analysis is briefly described in this section, in order to show the granularity of analysis on which the results are based. With video being a somewhat recent innovation in HCI research, tools and analysis techniques need to be developed and evaluated. The process described here was also used in Study 2.

The video-recordings ran to about eight hours, representing a considerable amount of data. The recordings were viewed in some detail straight through, each subject taking about two hours to transcribe. Protocols were not transcribed verbatim, but coded as key incidents and comments according to a classification scheme which was defined in part beforehand,

and in part evolved as the analysis progressed, as categories of observation were refined. An extract from a transcript is shown in Table 3.5.

Tape	Class-ification	Incident/comment
1020	query classific-ation	**classifies a query** HOW TO SPECIFY NEW LINK? **as an authoring query, compared to the ones done so far - so postpones query**
1090	browser cognition	Q: HOW TO NAVIGATE? - **looks at Criterion tree but realises** "*this is a 'Decision' type query*" **rather than Criterion based - goes back to list (only graph present)**
1208	browser strategy	**considers moving onto comparison queries, but decides to** "*do HyperCard properly*" **(complete all HyperCard queries first)**
1238	browser strategy	**requests Q/D/O/C for HyperCard, but as a graph -** "*the list became cumbersome for navigating - I originally got it for searching only*"
1285	Q O C notation	Q: EDITABLE AND SCROLLABLE? - "*confusing question*" - **it asks two Questions in one node**
1290	domain knowledge	**experience with HyperCard allows user to judge whether there'll be other relevant info** – "*I know nothing else is scrollable*"
1390	Q O C notation	**Do non-supporting Criterion object to an Option?** - "*not necessarily, but they aren't strong reasons for it - but there must be some relationship between the two...*"
1449	lists vs graphs	*"I had to start writing things down for lists, but haven't had to for the graph, yet"* – **working memory load?**
1460	query strategy	**surveying remaining two HyperCard queries - chooses one as it refers to info which has now been encountered**

Table 3.5: Part of an analysed transcript from Study 1

Key: **bold indicates analysis of the comment, or a question asked by the experimenter to the subject**; *italics indicates a comment*; UPPER CASE INDICATES THE NAME OF A QOC NODE.

This conveys the approximate granularity of observation which was used to produce a summary of the data, in which all observations of the same class were pooled. This collection of disjointed comments and notes was then substantially rewritten, with extensive reference back to the tapes where transcripts were unclear, and in particular, restructured around the task model which emerged from the data. A log of the browsers used for each task allowed one to see how subjects compared in browser strategy and proved a useful reference to see which browsers were active at any moment. In total, the process took about 1.5 months.

It is clear that video analysis can be a time-consuming business, given that the medium constrains the analyst to viewing all of the data in real time. Searching for incidents across seven tapes was most labour-intensive, and the development of video-analysis tools which digitise images and enable the transcript to be linked to the recording will revolutionise the process. On returning to the analyses after some time had elapsed, it became clear that whilst 'keypoint' transcription of the videos saves time initially, it also precludes re-analysis from a different perspective; a full verbal transcript allows the analyst to recover the context of incidents and comments. Clearly, however, the right granularity and perspective of analysis is dependent on the research. These are the costs of obtaining data which enable one to relive the experiment in a very real sense, and in particular, place oneself in the subjects' shoes – something which is most valuable in HCI.

3.2.2 A TASK MODEL OF QOC RETRIEVAL

From the video data, a large quantity of data was generated, from which initial pointers to the issues of importance emerged. A number of different and in some cases overlapping issues emerged, such as:

- characteristics of differently structured browsers;
- use of multiple browsers to arrive at solutions which one browser could not offer;
- development of cognitive maps of the representations;
- usability and role of QOC in the design process.

It was decided that an appropriate organising principle for understanding the data would be in relation to the *tasks* which QOC retrieval imposed. Observations from the data could then be discussed with respect to the tasks in the model which they influence. A model proposed after one study is necessarily incomplete, and will need refining as more research is conducted into QOC retrieval. However, it was assumed that there are four phases which the user must go through in order to locate information on a graphical browser, shown in Table 3.6.

Generic task	Experimental task
1. Select next DR query	Choose next QOC query to be answered
2. Translate sufficient information requirements into the DR notation	A target may be a combination of any of the following, depending on user's knowledge: a keyword, a QOC node of specific type and content, or a QOC substructure
3. Request DR representation from system	Work out which QOC representation(s) will present the target information most clearly, or allow that information to be inferred, and request from experimenter
4. Search representation(s)	Use knowledge of the QOC, and visually scan; if there is more than one representation, integrate information across them

Table 3.6: Generic tasks for retrieving QOC from a DR system.

The different factors which influence the performance of each task are also shown, which are explained within the relevant sections as indicated. This task model is illustrated schematically in Figure 3.8, which is now used as an framework to organise the results.

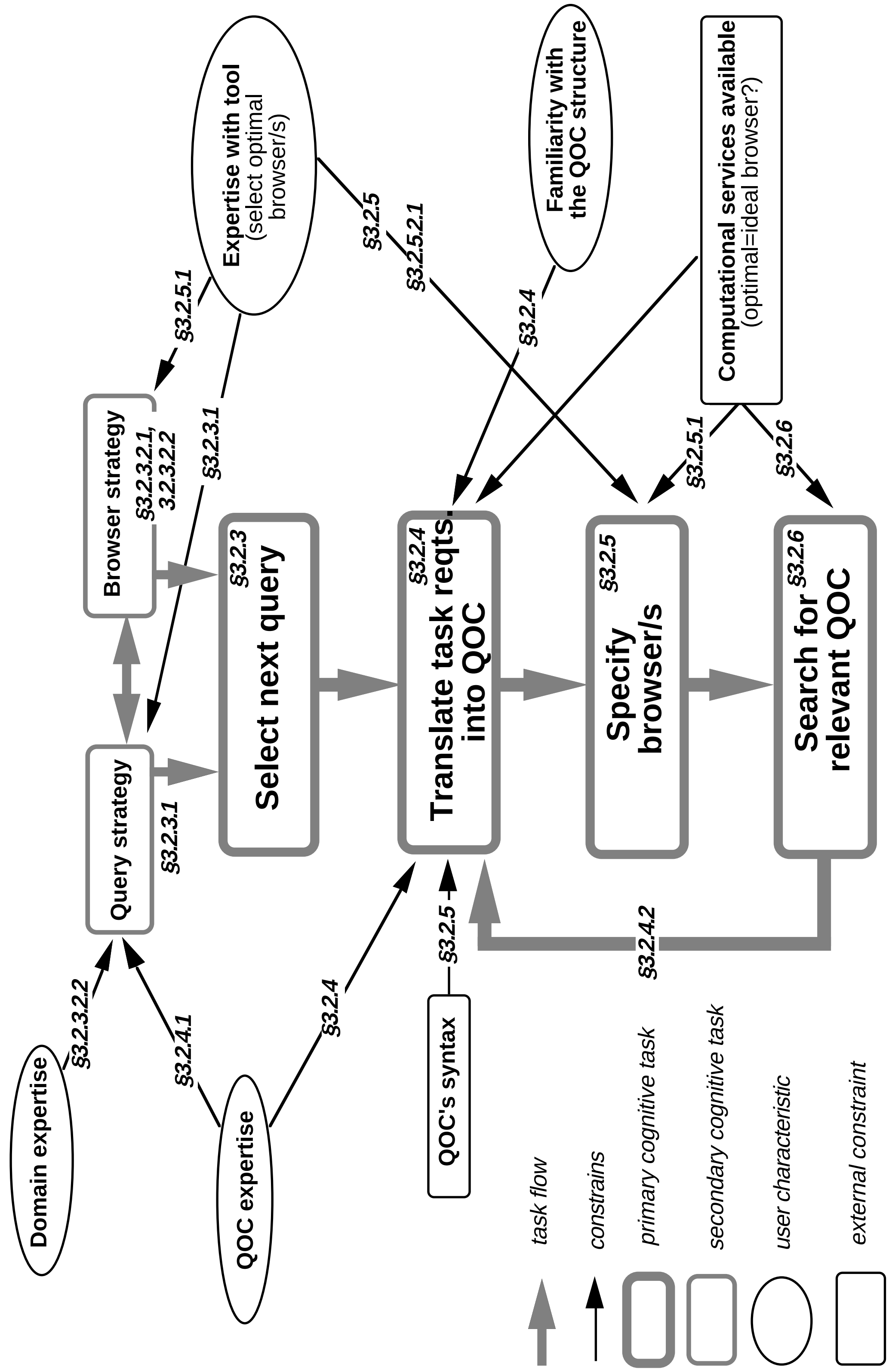


Figure 3.8: The task model of QOC retrieval around which results from Study 1 are organised.

3.2.3 Selecting the next query

Interest here is focussed on the factors which influenced the order in which subjects tackled the queries. A log of browser use and query scheduling is presented in Appendix

1. From the behaviour of the subjects in this study, the sequence of queries seems to be determined by two strategies: the *query strategy* and the *browser strategy*.

3.2.3.1 Query strategy

Query strategy refers to the way in which subjects conceptualise the queries they have – is there a logical order or pattern? As described in the method, subjects 1-4 were given queries ordered on a sheet. The listed ordering in fact reflected a query strategy of *deal with the next simplest query*. Deviations from this order were generally due to browser strategy [§3.2.3.2], rather than an alternative query strategy. Subjects 5-7, however, were left to formulate their own query strategy, which in all cases resulted in the classification of tasks according to the *hypertext system* concerned (NoteCards or HyperCard), i.e. *deal next with another query about the same system*. There were also examples of a more complex query strategy which combined the two: *deal with the next simplest query about the same system*.

3.2.3.1.1 Subjects' characterisations of query-difficulty

QOCs are essentially Question-oriented structures. A simple query therefore was:

How do users navigate in NoteCards?
(in the QOC somewhere) Question: How can users navigate?

Subjects had no difficulty with this sort of query. The problems arose with queries for which there were no specific Questions. Examples of such queries were those centred around the use of a particular Criterion or class of Criteria. This is difficult when only the usual QOC graphs are available, as a Question only occurs in *one place*, whilst a Criterion can be reused, and a class of Criteria will be *distributed* across a QOC.

Subjects commented on the difficulty of the queries in spatial terms (graphs being the primary representation used – §3.2.6.3.3). S6 observed that the simpler queries could be answered by going from "left to right" on the graphical browsers, that is, by following the argument from an initial Question, through the Options, to the Criteria, with the initial Question identified by keyword or topic matching. The harder queries, which generally involved finding Criteria, were classed as "right to left." S7 also referred to "left-right" and "right-left" queries (and also to "meta-questions" – questions about information distributed around a browser). S7 and a number of other subjects commented specifically that 'hard queries' were ones which did not map closely to the organisation of the graphs.

S2 expressed much the same view, although not in spatial language, commenting that easier queries were answered by 'primary Questions.' These were the initial Questions for the three subtrees in the QOC graphs, and hence the most visible.

3.2.3.2 Browser strategy

The second factor in selecting the next query is browser strategy, which refers to decisions subjects make, often quite explicitly, on two issues relating to browsers:

- the *number* of browsers they want to have open at any point
- the *level of detail* they want on a browser

3.2.3.2.1 Number of browsers

One way to cope with the problem of 'screen' (i.e. desktop) clutter and windowing problems, is to limit open windows to browsers which are being used to tackle the current query; all other browsers are closed. At the other extreme, subjects could keep open every browser they use, to facilitate rapid access to previously used information, rather than respecify the browser. The strategy adopted was a trade-off between accessibility versus clutter, both highly subjective concepts, which resulted in consistent differences between subjects.

3.2.3.2.2 Level of detail per browser

The second component which determines browser strategy is the preferred level of detail. In many of the queries, it was not necessary to use the full detail browser. The decision to use a particular browser rests on (i) the subjects' conception of what information is required, (ii) the overheads of requesting a filtered browser, and (iii) the importance which subjects give to having contextual information which is not needed to answer the query.

Figure 3.9 illustrates task and browser strategy in more detail than the initial task model (Figure 3.8), taking into account the elements just described.

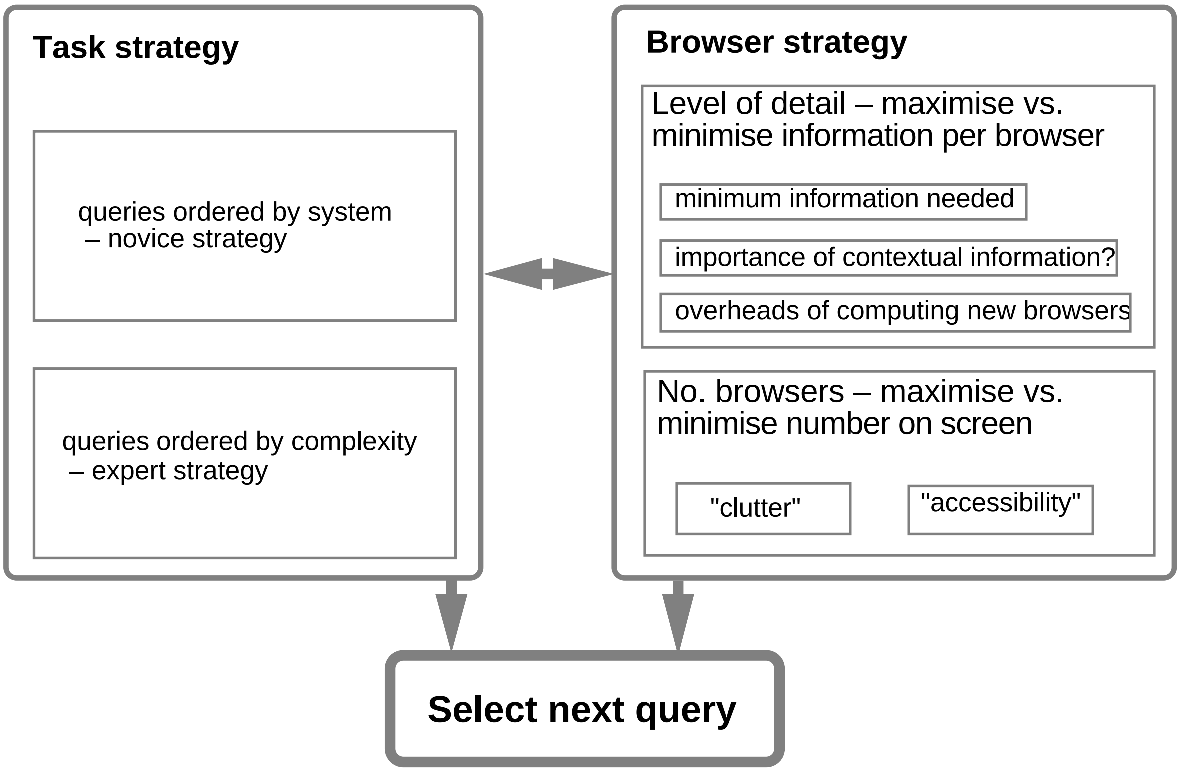


Figure 3.9: Elaborating the Task and Browser Strategy elements of the task model.

The following examples show how these strategies (highlighted in italics) were balanced by different subjects.

S1 declared a policy of trying to answer as many queries as possible with a browser before closing it. Early on with the simpler queries, he used browsers with slightly more detail than required, judging that being able to answer several queries from one browser outweighed the extra complexity of the graphs. For instance, graph-Q/D/O (Questions/ Decisions/Options) was initially requested, and subsequently used for another query. From then on, the most complex graph (Q/D/O/C) was used for the remaining queries. Whilst from initial observation, this subject would have appeared to be employing a system-oriented *query* strategy (*do all queries for one system first*), in fact it was the *browser* strategy (*minimise the number of different browsers used*) which was responsible.

S2 demonstrated a completely different approach. *For each query, a new browser or set of browsers* were requested, unless browsers from the previous query could be reused. Thus, browsers were nearly always of the *minimal detail* required to answer the current query. The query strategy followed the query list, and no attempt was made to plan ahead as S1 did. This ‘minimalist strategy’, in both number of browsers and level of detail, was not demonstrated by any other subject to such a degree.

S3, who used list representations only, was concerned to *minimise the number of browsers* he had to deal with, and except for one query, answered all other queries with the two most complex browsers available.

S4 demonstrated an interesting style, using *filtered lists* for the early, simpler queries, before moving onto the more complex queries, using *graphs* (his query strategy followed the sheet of queries). (Discussion of the benefits of lists compared to graphs can be found in §3.2.6.3.3). S4 was also interesting in that he *did not close any browsers* after use, simply stacking them to the side of the main 'screen' area; this meant that by the end of the tasks, he had nine browsers open, in three different formats (list, graph and Criterion tree).

S5 (who was given shuffled cards with queries) organised queries according to whether they were *NoteCards, HyperCard, or both* (i.e. comparisons between the two). A further judgement of difficulty was made about the single-system queries, where *queries which mapped closely to the graph structure* were judged to be easier [see §3.1.3.5 for ratings of query difficulty].

S6 followed S5 in *organising the queries by system, but not by difficulty.* This led to a complex second query, which required a full detail browser. Subsequent queries about NoteCards were then answered using this browser, and on moving onto HyperCard, he requested the most detailed browser immediately, presumably because the strategy had worked for NoteCards. Queries comparing systems were answered after queries about individual systems, to "get a feel" for each before comparing them. In keeping with his strategy of maximising the information per browser, S6 requested full detail browsers and Criterion trees before studying any of the comparison queries in detail. He felt that full detail was not too complex, and that given the size of the desktop the extra browsers could be easily handled.

S7 utilised a complex query strategy similar to S5's, which classified and ordered queries according to whether they dealt with *NoteCards, HyperCard or both,* but within each of those groups, went a step further and decided to *tackle 'easier' queries first*. HyperCard queries were addressed first, as this was the more familiar system. (It was noticeable how domain knowledge added confidence to users, for instance computer scientists expressed some uncertainty at tackling queries with a strong psychological component, deferring such queries until the end).

S7 asked for the most complex graph to begin with, adopting an initial browser strategy of trying to *use only one display at a time,* which he could get to know. He was prepared, however, to change if necessary. When he moved onto NoteCards, the full-detail QOC graph was again requested, as he was satisfied with this level of detail when dealing with HyperCard. However, towards the end of the queries, S7 requested the minimal detail

(Q/D) graphs, as he felt that he had familiarised himself with the QOCs. He then remarked that it would have been useful to have used the Q/D graphs earlier, due to their clarity.

3.2.3.2.3 Summary of query sequencing behaviour

From studying these different query sequencing behaviours, a number of interacting factors have been identified which subjects balanced in different ways.

Five subjects showed little tendency to use filtered browsers, one (S6) made use of simpler ones on occasions, and one (S2) did so consistently. There was a strong tendency amongst the subjects who received no prescribed order of queries to adopt a query strategy which avoided switching attention between the two systems of interest, and all subjects, regardless of query strategy or preferred level of detail, reused browsers for the next query when possible.

It is likely that if subjects were going to follow a query strategy other than simply following the query sheet, then the *system*-oriented strategy was the most obvious one to adopt (e.g. *do HyperCard first*). Assessment of queries on the basis of the number and type of browser required (a *browser*-based strategy) is likely to be far too complex and time consuming for novices. Subjects focus primarily on answering the query; making most efficient use of browsers is a secondary consideration. The implication is that as they acquire more expertise with both QOC and the tool, designers will generate more sophisticated work plans leading to better schedules, taking into account (i) *which browsers* are needed for a task, (ii) the benefits of using a *tailored as opposed to general purpose (i.e. sub-optimal) browser* for a task, and (iii) identifying those *queries which are dependent on others*, by *reusing* browsers and information gained from earlier tasks.

3.2.4 TRANSLATING REQUIREMENTS INTO QOC NOTATION

In order to communicate with a DR system, the designer must at some point translate aspects of his requirements into the DR notation. Depending on proficiency with the notation, this can be done more or less efficiently. A 'good' as opposed to 'impoverished' expression of requirements correctly specifies several elements which QOC relevant to the query should contain. This can be illustrated if we take two of the queries used, and compare hypothetical examples of good and impoverished translations:

How does a NoteCards user specify the destination for a new link, and why was this method chosen?

Good translation (specifies nodes and relations):

"I need a *Question* about *how links are created*, which may have *supporting Criteria*, about things like *intuitiveness* or *ease of use*."

Impoverished translation:

"I need nodes about navigation, and reasons for choosing that method."

A more complex query was: *How do the systems compare in the extent to which their designs took into account the cognitive psychology of users?*

Good translation (specifies nodes and relations):

"I need examples of *Decisions, supported by Criteria* which are about things like *memory, attention* and *learnability*."

Impoverished translation:

"I need QOC which combines cognitive psychology with design."

The ability of designers to translate their requirements into succinct statements is not dependent solely on their familiarity with QOC. The above examples of well-specified QOC descriptions are only realistic if the designer also knows *how the QOC is structured*—as in all communication, the sender must have a model of the receiver in order to formulate the message. If the designer has no concept of how the QOC has been organised, then his ability to formulate structural queries which map to the stored representation is impeded. The value of QOCs (and Criterion trees) which have been organised along some agreed structure is thus obvious—the designer can be sure that there will be information under particular key headings. Due to the semi-formality of QOC, the same piece of information can be represented in different ways. The 'overlap' in the conventions adopted by the designer and the QOC being searched can be further increased if they share an understanding of principles for *well-structured* QOC, as proposed later [§10.3].

3.2.4.1 Analysing a QOC translation error

Errors can be operationally defined for the translation, specification, and search tasks. *Translation* errors involve the incorrect translation of information requirements into QOC (e.g. representing alternatives as Criteria). However, as noted, QOC's informality means that sometimes ideas can be represented in a variety of ways (e.g. phrased as an Option or a Question);[3] in these ambiguous situations, an error can be defined operationally as the

[3] Study 3 presents many examples of this, e.g. §5.3.1.2, §6.2.4.

translation of requirements into QOC which does not match the form of the existing QOC (see example below).

By way of illustration, consider one instance of an error in the use of a browser to answer a query. In the transcript below, S5 tackles his first query,

> Ok, *(reads query)* "What ways were considered for viewing material in NoteCards?" — so this would be something in the QOC which has a Question which says something like..er... 'how is material to be shown?' *(requests NoteCards graph of Questions and Decisions, and finds Question on navigation)*...well what were considered were (reads out set of Decisions) 'graphical browser, history list, clicking buttons from card to card, and find-operation' – is that right?
>
> *E: Why did you ask for this browser?*
>
> I want to find the point where the designers considered how they might allow the user to view the material, so there should be a Question somewhere on how should material be presented to users...ah! *(spots correct Question on browser)*... 'How should cards be viewed?' I wouldn't expect to find that under 'How should a node be conceptualised?' *(the parent node of that subtree)*...that's more about the model of nodes the user should have, which is a different question...anyhow, the ways which were considered were...there was only one way considered *(reads out the Decision)*: 'One of any number of windows'...
>
> *E: You've only got Questions and Decisions there.*
>
> Ah...so that's the *Decision* which was made...so I need the Criterion...the..er..Options.

S5 makes two errors immediately. Most seriously, he confuses Decisions with Options, and so requests a QOC filtered down to Q/D, when in fact he needs access to the Options as well. Secondly he interprets *viewing material* as *get to material*, rather than the more literal interpretation in the QOC, which refers to the windowing system to be used. This confusion is resolved after a while.[4] After a period of deadlock, the experimenter prompts the subject with a question, and whilst he understands the meaning of the query, S5 still fails to realise the error and expresses surprise on finding that only one 'Option' had been considered in the browser he has requested. In the end, the experimenter hints at the problem, and S5 corrects himself, although confusion over the different node types reflects his inexperience with QOC.

In terms of the task model, errors in browser specification arise from two sources: either incorrect translation into QOC notation [§3.2.4], or lack of expertise in using the browsers offered by the QOC tool [§3.2.5]. In the above example, S5 made his error due to a translation breakdown, in that he did not discriminate correctly between an Option and Decision; browser specification was executed correctly – he obtained the structure which he wanted – but that structure was wrong. It can be seen therefore that this error is a *mistake* because the *intention* was incorrect. In terms of Norman's (1986) generic model of user-system interaction, the *goal* was incorrect, but the *specification-execution* stage was correct, so that the *evaluation* of the results (the new browser) failed to reveal the

---

4 See Study 3 [§6.2.3, §6.2.5] for further evidence of entity-naming problems.

problem. It was not until prompted by the experimenter, that the goal was reformulated (i.e. re-translated into QOC: "I need the *Options*"), and the goal-result disparity was identified.

There were eight other instances of translation errors. The errors logged in Appendix 1 are summarised in Table 3.7.

Subj.	Query	Information requirement	Browser used	Error
1	5	reasons for a specific decision in NoteCards?	NoteCards Criterion Tree	missing Decisions to which to relate Criteria
1	7	reasons for missing functionality in HyperCard ?	HyperCard QDO	missing Criteria
2	4	what HyperCard alternatives considered for a particular problem?	compares NoteCards & HyperCard QDO	NoteCards QOC irrelevant
2	15	what NoteCards functionality is available for particular tasks?	HyperCard Criterion Tree	HyperCard irrelevant
3	3	what NoteCards alternatives considered for a particular problem?	NoteCards QDC	missing Options
4	10	how do systems compare in use of Criteria of a certain class?	compares NoteCards QDO, QDC, & HyperCard QDC	Criterion Trees needed; impossible to compare Criteria between graphs
5	3	what NoteCards alternatives considered for a particular problem?	NoteCards QD	missing Options
5	7	reasons for missing functionality in HyperCard?	HyperCard QDC	missing Options
6	2	how to perform standard operation in NoteCards?	NoteCards Criterion Tree	missing Questions and Options on Criterion Tree

Table 3.7: Summary of errors in translating requirements into QOC.

3.2.4.2 The *process* of translating into QOC

It is not always the case that requirements must be translated into QOC completely as a free-recall task (i.e. without any cues). A number of incidents indicated another way in which subjects derived QOC descriptions of requirements.

Mayes et al, (1988) demonstrated that even experienced users could not recall the names of menu-bar and menu items in a familiar application, but experienced no problems in actually using the system. The conclusion was that performance was dependent on the presence of the display, whose structural and semantic cues guide the user's activity as the task progresses. Something akin to this use of available screen cues was observed in requesting QOC browsers. In translating requirements, subjects made use of the existing information on screen, in a sense'bouncing' ideas off open browsers as an aid to refining in their own minds the description of the browser(s) required.

S5, for instance, was dealing with a relatively complex query which required the most detailed graph, plus a Criterion tree. Left open on the table from the previous query was the simplest graph (Q/D). This was at the top of a pile of several overlapping windows. He studied this browser initially because "it was there," on doing so quickly realised that it was lacking certain information like Options, and switched to a more complex graph (which was in fact also open, but hidden by other windows).

This strategy of 'look, then think' also seemed to be at work with S6. Having considered a query, he looked at the currently active browser, a Criterion tree, and realised almost immediately that the query was "more of a Decision one", that is, the format of the query mapped more closely onto a graph which had Questions specifically on the topic of interest (navigation).

This illustrates the limitations of a purely sequential task model. There are two possible ways in which the task model could account for the above incidents. The first is that subjects did not attempt to translate the query before processing information from the world. The process *started* with processing external information – the *search for relevant QOC* stage in the model – which then drove the translation. Alternately, an underspecified QOC *translation* of requirements may have formed from reading the query, but insufficient to explicitly *specify* a browser. Bypassing this stage, subjects moved directly to *checking* the available browsers. Figure 3.10 shows a subsection of the task model originally presented in Figure 3.8, with these two processes.

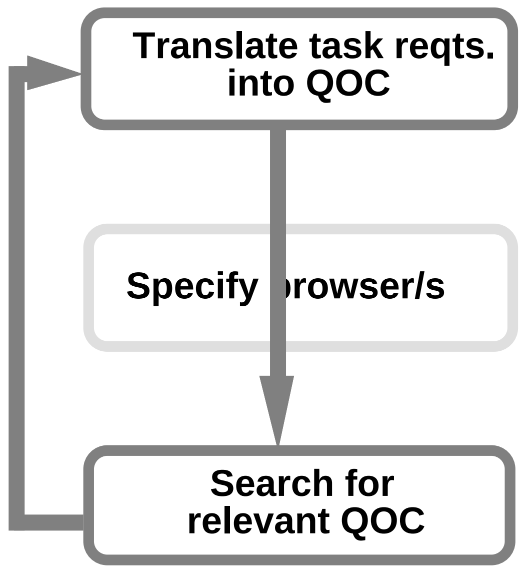


Figure 3.10: The role of feedback from external representations on translating information requirements into QOC.

The above incidents illustrate the importance of opportunistic, situated action as a product of interaction with information in the environment (Suchman, 1987). This use of the 'information environment' as a cognitive aid is also captured in concepts associated with 'distributed cognition' (Hutchins and Norman, 1988; Norman, 1988; Olson, 1990). This approach recognises that in the real world the functional capacity of our cognitive resources (e.g. working and episodic memory) is expanded through the use of the 'cognitive artifacts' (Norman, 1991) with which we surround ourselves in order to construct tailored work environments (e.g. post-it notes, diaries, sketches, simulations).

Finally, opportunistic use of information in the environment creates problems for predictive modelling in HCI. Carroll and Rosson (1987) point out that one cannot always assume that the user always starts with a 'Goal' specified at the appropriate level (in this case as a QOC representation ready for specification to the system). Users are notoriously unpredictable, adopting different strategies, and not always rational. It is possible that if the users were experts (i.e. more predictable), then rules could be formulated as to when and how available browsers could be used. Payne (1991) describes display-based TAG (D-TAG) as one approach to modelling interaction in which the user is driven moment-by-moment by the information display (e.g. menu-navigation).

A predictive cognitive model of browser specification would also have to take into account the effect of the QOC system's functionality on how users translate requirements into QOC. Those factors are considered next.

3.2.5 SPECIFYING BROWSERS

3.2.5.1 The constraints imposed by the notation and the tool

Queries can be classified according to where they fall in degree of match between required information and browser structure. The range of possible representations which can be computed over a QOC is a function of two variables:

- the characteristics of the QOC notation;
- the functionality of the QOC representational environment.

'Notation' is used here to refer to the syntactic constraints of the language. Thus, in QOC, *Questions* lead to *Options*, which are *supported* or *objected to* by *Criteria*. One characteristic of a notation, therefore, is the set of *syntactic constraints* which it imposes on structure.

As is illustrated shortly, notational constraints are orthogonal to a system's capacity to compute novel structures within those constraints. It is not always the case that once the designer has decided what QOC structure would answer his query, that this is available from the system. It is necessary to take into account situations where the 'ideal' QOC representation which would meet information requirements perfectly, cannot be displayed by the tool, due to its limitations. In this section, the distinction is made between the *ideal* QOC structure which would most concisely answer a query, and the *non-ideal* browser or set of browsers which the QOC tool can offer.

In this study, users were only offered four levels of detail, based on a core QOC graph structure, and four levels of list. "Expertise" therefore, in the context of using a browser driven QOC system, implies not only the ability to translate from task requirements to notation, but to then optimise the system's facilities by selecting representations which will maximise the chances of finding relevant QOC.

The 'system' used in this experiment was clearly impoverished. Being paper-based, only prepared 'canned browsers' were available to subjects. Let us now consider an example where information across non-ideal browsers had to be integrated. One query imagined that the designer is aware that NoteCards was developed to support particular kinds of tasks, and wished to find out how this had shaped its design:

NoteCards was developed to support specific sorts of tasks. Identify three decisions in its design which were made specifically with these tasks in mind.

This is quite a complex query, centred around the tasks NoteCards supports. In QOC, these are usually represented as goals to satisfy, i.e. as Criteria. Consequently, the best starting point was the NoteCards Criterion tree, to identify task-related Criteria.

Relevant Criteria could be found under the general Criterion *meet task requirements.* Subjects needed to find Criteria displayed in bold (i.e. which supported final Decisions). Figure 3.11 shows a fragment of Criterion tree, with three bold Criteria: *represent concepts as concrete screen objects*, *provide map level knowledge of structure* (both used to support the decision *graphical browser*), and *support backtracking* (which supported the decision to use *history cards*). Decisions supported by these Criteria then had to be located on a QOC graph, as shown. In this example, all of the Criteria are clustered together on the graph – in other queries they were dispersed across the structure.

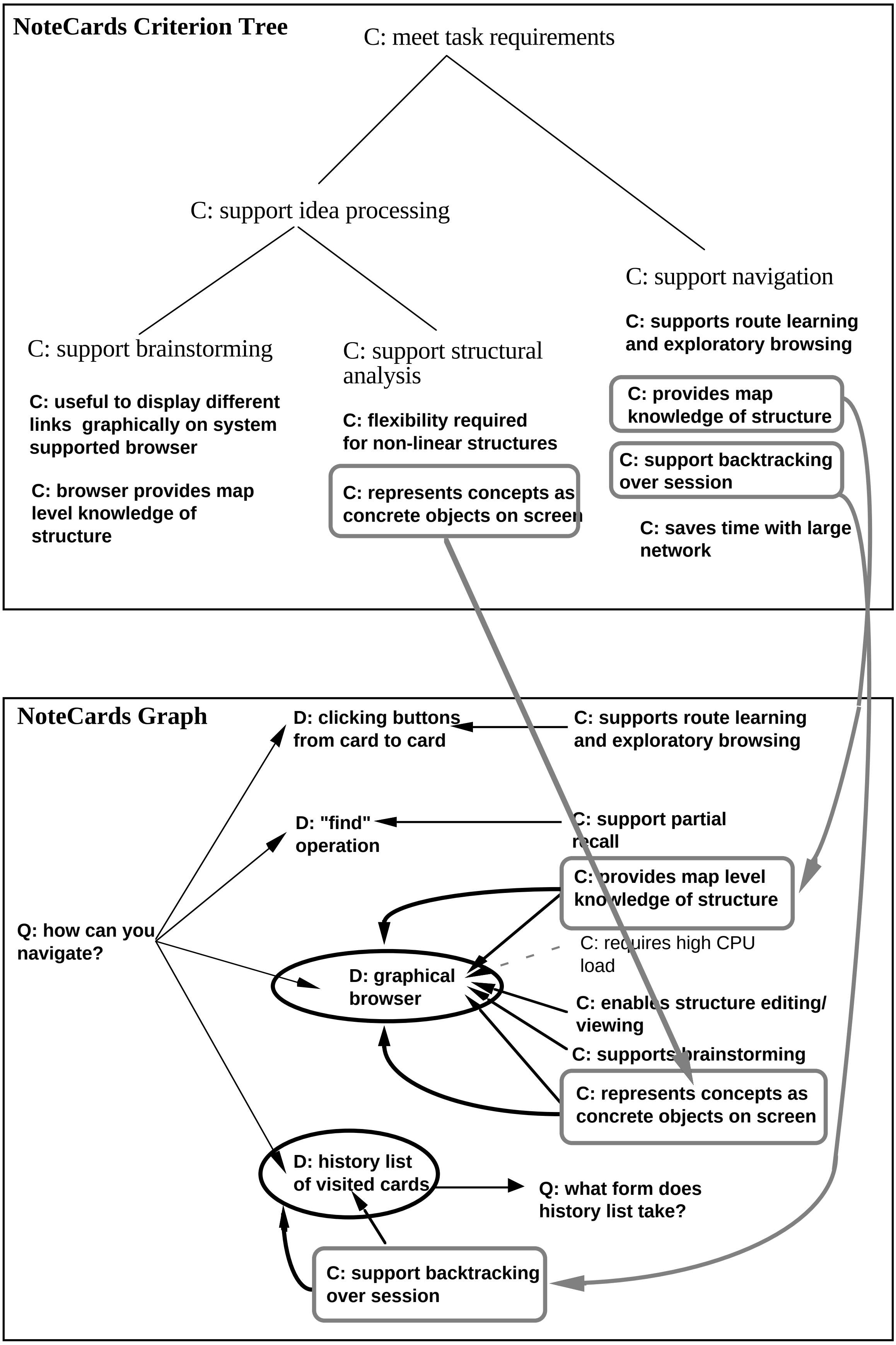


Figure 3.11: Mapping between a Criterion tree and QOC graph. Subjects had to establish these mappings mentally, because there was no representational support.

It was the task of integrating information across the two browsers which created the highest cognitive overheads; without tool support, subjects had to construct their own representation of the relationship between the two browsers, that is, *decisions supported by task related Criteria*. The *ideal* browser in this situation would be one which made this relationship explicit as a new graph, for example, Figure 3.12.

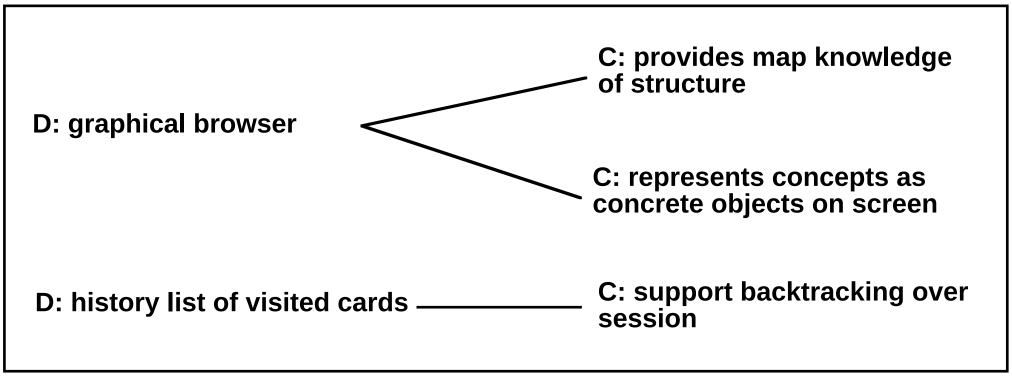


Figure 3.12: Answering a query using a single representation, which makes explicit the relationships between a QOC and a Criterion tree (summarising the mapping shown in Figure 3.11).

The intersection between the Criteria in the Criterion tree and their relevance to the QOC is now explicit rather than implicit. This illustrates how the tool's functionality is independent of the QOC notation. Although the user is able to specify the QOC which would *ideally* solve the query, he must instead use the non-ideal browsers. The more expert a user is with a QOC system, the more able he is to utilise non-ideal representations to infer new information.

Two interesting incidents arose in which the ideal browser was in fact the Criterion tree (i.e. the ideal browser was available), but rather than using it, subjects tried to generate mental representations of parts of it (S3 forgot the Criterion tree was available, and S4 made the erroneous decision that it was of no use for that query).

Let us consider S4, who was attempting to answer the query about how different tasks were supported . He found several Criteria in the QOC, *support brainstorming, support partial recall,* and *support route learning.* Whilst trying (unsuccessfully) to make sense of these, he commented:

> I would not expect to be able to answer all these questions using this browser alone.

He went on to observe that there was no way of telling *the relative level of abstraction* of these Criteria. For instance, was *support brainstorming* an end in itself (i.e. one of the main tasks he was searching for), or a bridging Criterion to a more general goal such as *increase group creativity*? S4 was forced to infer a hierarchy of Criteria when he should have been able to access the original designers' view:

> All the onus is on my inferences relying on my knowledge of HyperCard ... rather than on getting support from the designers' domain knowledge which should have been captured by this design document (the QOC).

When shown the HyperCard Criterion tree, S4 immediately recognised the useful information which it offered, but also wondered how he was supposed to know which classifications of Criteria could be found there.

To summarise, two factors influence browser specification:[5]

1. **QOC system's constraints**: How sophisticated is the hypertext browser facility – is the ideal representation available?

2. **Expertise with QOC system**: how efficiently can the designer use the available representations to answer his query? Note that knowledge of the system's limitations will also affect which browsers are requested for a given query, i.e. the browser strategy [§3.2.3.2] .

3.2.5.2 Errors in specifying browsers

The error classification work by Reason (1984) and its application to HCI by Norman (1984) views behaviour as intention-formation followed by execution, which allows one to distinguish between *mistakes* and *slips of action*. The former are due to incorrect intentions, and the latter to faulty translation into appropriate communication with the system/world. Either or both of these can be responsible for an error at any of the task stages. Errors in some stages will invariably be *mistakes*, because the *goals* set (i.e. intentions) are contingent on the correctness of output from earlier stages. Within the task model, incorrect browser specification is driven by the translation activity, which sets the goal, that is, such errors are usually *mistakes* (illustrated in the example below). *Action-slips* are errors in the mechanics of requesting a browser – in the context of this study, this might take the form of a verbal error when requesting a browser.

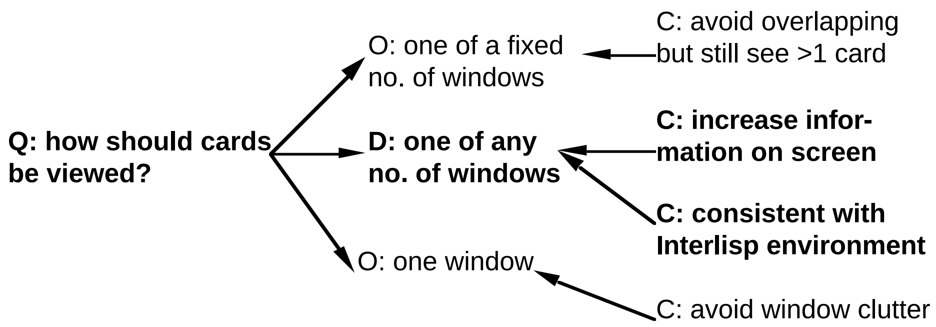
Errors in the execution of this task phase are defined as requests for inadequate representations when there are others with sufficient information. This covers requests for inadequate browsers whether they are available or not (the minimal solution paths for each query are shown in Appendix 1). There were two requests for browsers which were classed as errors (Table 3.8); it also happened to be the case that these browsers were not available.

[5] Of course, if the translation of requirements into DR is too inaccurate to begin with, the retrieval process will inevitably end in failure.

Subj.	Query	Information requirement	Browser used	Error
4	3	what NoteCards alternatives considered for a particular problem?	NoteCards QC	missing Options (and browser unavailable)
6	3	"	NoteCards DO	missing Questions (and browser unavailable)

Table 3.8: Summary of errors in specifying QOC browsers.

Both occurred for the query, *What ways were considered for viewing material in NoteCards?*, in which the answer was to be found in the QOC structure shown below (it would seem to be coincidental that one query was responsible for both errors):



S4 erroneously requested Questions & Criteria for NoteCards, which would not have helped, because Options were needed in order to answer the query.

S6 requested Decisions & Options graph for NoteCards, which although containing the relevant Options, lacked the crucial Questions to group Options together in context, making it impossible to identify those Options which addressed the query. It is likely that requests such as S6's are a common feature of early use of QOC retrieval; on reading the query, the primary information requirement was correctly identified as *Options*, but the second element needed in formulating a QOC query is the *Question* – what Question do the Options answer? Using a structural search pattern which includes a description of the *Question* (i.e. the context) in which Options should occur, is more powerful.

3.2.6 SEARCHING FOR RELEVANT QOC

Once subjects had understood the query, translated it into QOC notation, and obtained the browser(s), they had to identify relevant QOC within those representations. Errors associated with this task stage include visually scanning over and ignoring target structures, misunderstanding nodenames, assuming that all relevant information had been found when there was more, and incorrect cognitive maps of the structure leading to navigational and conceptual problems. The effectiveness of visual search depends on surface level factors such as clarity of structure and meaningfulness of nodenames, and on deep level characteristics like the QOC's organisation. The following sections deal with both of these levels.

3.2.6.1 Keyword search

The usual way to locate a node was via keyword search; a word or phrase would be lifted or paraphrased from the query, and, depending on how that information had been translated into QOC, the graph, list or Criterion tree would then be searched for relevant Questions, Options, Criteria, or combinations of these.

Visual search is useful for small, localised searches, and is clearly necessary when it comes down to actually scanning the display for relevant QOC. Needless to say, it, is limited in power for searching large structures, and a QOC tool would have to support automated string searches as one form of navigation.

3.2.6.2 Structural search

Halasz (1988), in his agenda of issues for hypertext research, listed *structural search* as one of the requirements. This refers to the possibility of specifying not only fragments of the contents of one node (i.e. keyword search), but the relationships between two or more nodes. Clearly, such a facility would be immensely useful for locating QOC structures.

S5 suggested that it would be useful to find 'deciding Criteria', that is, Criteria which overruled others in decision making. In fact, these Criteria were marked on Criterion trees in bold, but given the use of paper-based representations, there was no way of using these as an index into the main QOC browsers [see §3.2.5.1 for discussion of the overheads which doing this 'manually' caused, and how to support mapping between representations]. The advantage of structural search facilities, however, is that an infinite variety of virtual structures can be made explicit, so that the user is not confined to working with generalised representations. *Deciding Criteria*, for instance, could be refined into deciding *hardware* Criteria, which took effect in relation to *decisions about graphical issues.*

Structural search templates for computing structures such as this (on a hypothetical system) are shown in Figure 3.13.

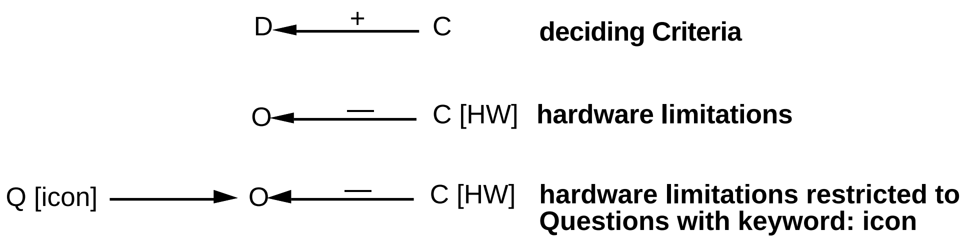


Figure 3.13: Structural search templates for specifying a range of increasingly complex virtual QOC structures.

Consens and Mendelzon (1989) have described a system which supports structural search facilities via an interface in which queries can be formulated by directly constructing graphical templates similar to the above examples.

3.2.6.3 Spatial cognition and browser design

In a multi-browser environment, the comparison of different representations for matches and mismatches is likely to be a frequent mode of working. Observations in this study confirm the importance of the spatial cues afforded by different representations.

3.2.6.3.1 Spatial consistency

The importance of *spatial consistency* became clear. On most tools, when graphs are filtered, the 'space' left by nodes which have been removed is filled by surrounding nodes, as the algorithm does not preserve absolute screen positions. It would, however, be extremely useful if users could choose to preserve position, so that old nodes were simply deleted, leaving others unaffected. As a consequence of the way in which the graphs were structured (using layers in a graphics package), spatial consistency was preserved across different levels of detail, which subjects found most useful.

The preservation of space has a number of valuable benefits. Whilst the presence of four or five different representations which partially share elements can be overwhelming, if the user can be cued as to how one relates to the other, the disorientation is reduced. The development of cognitive maps relies on a set of consistently placed landmarks, such as root nodes, or particularly salient structures. It can be reasonably hypothesised that a set of graphs which preserves spatial consistency will be less disorienting than a set which leaves no spaces—space is a crucial element in spatial environments.

An example of this was found with S3 who used lists only, and found that the similar list structures helped make comparisons between the two systems. However, in filtered *lists* (e.g. Q/D), the space was closed up where material was hidden (this was an unintended difference between graphs and lists). Whilst this reduced the size of the list overall, potentially informative spaces were lost. Taking into account the impoverished spatial cues which a list already has in comparison to a graph, this hid the hierarchical organisation even further, to the extent that when asked at the end of the experiment what the QOC's organisation was, S3 was not aware that there were three sub-hierarchies.

S7 commented that it helped to have similar HyperCard and NoteCards browser structures. Since spatial knowledge *transfers* between browsers, S7 (and other subjects) were able to quickly locate nodes within QOCs which they had never seen before.

A second characteristic of space is that it is *informative.* If a user expects to find an object or structure (e.g. some Criteria) and finds an empty space, it instantly communicates a problem. One instance of this occurred when S5 requested a browser to find out why a particular design feature had not been included, went to the region of interest, and found a large space where he expected to find several design alternatives; this immediately cued him to realise that he had forgotten to request Options. Whilst he would probably have realised his mistake in due course using another representation, there is no doubt that structural spatial cues highlighted the discrepancy in content.

The development of *cognitive maps* of the browser structures is crucial to their effective use (the relationship between spatial cognition and hypertext is examined in more detail elsewhere – Shum, 1990). An example of the development of structural awareness was with S2, who explained that he remembered a node's location, because when he had come across it earlier whilst performing another task, he had made a mental note that this was where he had expected to find it. It is likely that this background knowledge (*this is where it should be*) served to enrich the coding of location, meaningful spatial/conceptual organisations being more easily memorised than meaningless ones.

Spatial consistency: critique

When is it good to preserve spatial consistency? It is clearly desirable when browsers are to be compared for differences in structure and content – subjects needed to be able to compare browsers, an important component of which is the presence and absence of spaces.

Spatial consistency is also significant when a given browser is modified, for instance by bringing in new link and node types, and removing others. Whether the user intends to compare them, or simply use the new browser alone, orienting to the new display is likely to be aided if nodes which carry across from the old one are preserved in the same relative position to each other.

Spatial consistency is not always useful however. Firstly, the resulting display may in fact be harder to understand than a more conventional graph, as the organisation may be less obvious. This is especially likely to be the case for users who are not familiar with the browser from which the second one was derived; in other words, spatial *consistency* is a context dependent property (consistent with *what*?)—it is only intelligible if one is familiar with the source representation on which it is based.

Another limitation is when a new display is specified which represents a completely different perspective of the data. If the user computes a new browser which shares few or none of the nodes in an existing browser, the overall graph is inevitably more coherent if it uses a fresh layout, rather than using an existing browser as its basis.

3.2.6.3.2 Cognitive maps of the QOC structure

As spatial cognition research has confirmed, we form cognitive maps of spatial environments through a variety of channels. How did subjects come to acquire maps of the QOC structures through their use of 2D lists and graphs? All subjects were asked at the end to described the structure of the QOCs they had been using.

Nearly all were able to correctly report that there were three subhierarchies, which dealt with Questions about *nodes*, *links*, and *navigation*, respectively. However, two points of interest arose. Firstly, S3, who was only presented with *list* browsers for the whole experiment, was aware at the end of the hour only that information on navigation was localised in one area, but not that it represented one of three subtrees.

Secondly, at one point, S6 had no idea of where to find navigational information. It may be that this was due to his extensive use of lists, as opposed to graphs (cf. S3). However, when asked at the end (having used graphs by then) how the QOCs were structured, S6 described the three subsections as *presentation, authoring,* and *navigation,* clearly demonstrating an awareness of the three clusters. These descriptions are understandable to some degree, as the subtree on nodes dealt with some presentational issues (e.g. card metaphor; windowing) and the links section dealt largely with the design and use of buttons, primarily an authoring activity). Consistent with this, S6 referred to queries about link creation as 'authoring questions.' These examples illustrate quite clearly how the same structure can give rise to different cognitive representations; whilst this study was an inappropriate experimental design to make strong claims about causality, there is evidence to support the claim that, compared to graphs, the development of cognitive maps was slowed by the extensive use of lists.

3.2.6.3.3 Lists versus Graphs

This section describes the problems encountered with lists, as well as advantages which were found.

Graphs were preferred to lists by all subjects. S1 preferred them both aesthetically (text too dense on lists) and functionally. He found it easier to see multiple connections, and after initial success decided early on to stay with them. S4 considered that graphs gave him more information per unit area than lists.

A list was requested by S2 experimentally, to see how useful it was, but he found that it hid the structure. He then requested the equivalent graph, and reported that matching from query requirements to browser was easier with a graph.

S4 requested a list for the first query, a relatively simple search task for a specific Decision. Lists were used for other simple fact-finding queries for each system. When asked why lists were used, S4 explained that they were small enough to allow rapid location of well defined information, but he switched to graphical browsers for more complex queries which asked "how and why" questions. S4 later commented that graphs more clearly showed interdependencies.

In a search task for known targets (four Criteria), S6 first requested a graph; on seeing its complexity, however, he then requested the list representation, as it would be easier to tell how much he had covered in the search compared to a graph. A list format Criterion tree was also used on another occasion, again because he felt that linear search was faster. S6's reasoned that searching a list involves scanning more or less in one dimension only (i.e. vertically), whilst scanning a graph adds the horizontal dimension. There are however a number of trade-offs potentially in favour of graphs:

- the overall structure may be easier to convey graphically than as indented text, although if the structures are large (e.g. more than about 20 Questions) tool support is needed;
- high level views of the QOC with a wire-frame for rapid scrolling, as provided in gIBIS and NoteCards, may be a useful navigational mechanism which text does not usually offer (although a collapsable outline has some similar properties);
- multiple relationships (e.g. between Options and Criteria) are more easily shown graphically (although a matrix can often be as effective);
- however, if a particular *substructure* is being searched for, graphs would be expected to be better, both visually, and for formulating queries for a structural search facility [§3.2.6].

To summarise observed behaviour with lists and graphs:

- lists were preferred by some subjects for *well specified search tasks,* as elements are roughly aligned down the left margin(indentation was small), whereas graphs' nodes are spread across two dimensions;
- lists were *aesthetically less pleasing* than graphs: subjects who volunteered a preference always preferred the graphs, one reason being the density of text on lists. This may also have caused several subjects to miss nodes for which they were searching;
- lists may have *hidden the macrostructure* of the QOCs, impeding the development of cognitive maps of the structure.

One conclusion is that developers of QOC should not limit themselves to graph-based representations. Hierarchical lists have a role to play, and in hypertext form, can be used to convey multiple 'tables-of-contents' views (Carey et al, 1990). Other representations such as interactive matrices (VanLehn, 1985; MacLean et al, 1989; Lee, 1990) convey certain kinds of information better than graphs, particular large scale trade-offs between dimensions (e.g. the Assessments of seven Options by ten Criteria becomes confusing when represented as a graph).

3.2.7 EXPLOITING FILTERED REPRESENTATIONS TO THE FULL

As hoped, a number of subjects made use of filtering as a means of reducing visual complexity. Use of filtering depended largely on a subject's browser strategy [§3.2.3.2]. If they favoured minimal levels of detail for a given task, then browsers tailored to the query were used. For instance, in comparing Decisions between two systems, S2 used the minimal graphs (Q/D), which made the visual comparison much simpler (see Appendix 1 for a full record of subjects' use of browsers). When searching for a Question which he knew was present, S5 replaced browser Q/D/O/C with Q/D/O, and found the Question immediately, commenting on how much clearer the display was.

S6 commented that the graphs varied in a uniform 'all or none' level of detail, i.e. *all* Criteria removed, or Options shown for *all* Questions. A fisheye view (Furnas, 1986) would be useful, to examine in detail one particular node or area, whilst keeping the others at minimal detail. One could imagine QOC fisheye views over a range of functions:

full detail for all Questions consequent to the selected Question;
all Questions dependent on the selected decision;
all decisions supported by Criteria in the same class as the selected Criterion;
full detail to a depth of two Questions from the currently selected one.

An additional suggestion for filtering was made by S4 (after completing the task in Study 2). It was suggested that there should be some way in which all the QOC elements associated with a common topic or task can be treated as a group for filtering purposes. Some examples which illustrate the scope of such a facility are:

all Options which apply if Company X deliver the promised chip on time;
all Questions and Options which came to mind following Jim's suggestion;
all Criteria and Assessments which were added during Project meeting 7.3.

A grouping facility such as this is task-oriented. Just as similar virtual groups can be computed by combinations of date, author, or keyword, grouping by task would require a *task* attribute, which although ignored most of the time, could be utilised for subsets of QOC for which subsequent retrieval is judged to be particularly helpful. Thus, taking one of the above examples, one can imagine the designer opening a session by declaring that until stated otherwise, all modifications to arguments were to be tagged by the system with *task* attribute *"if Chip X arrives on time."*

Although filtered browsers were used to some extent, they were not used as much as hoped. The reason many subjects were content to use the most complex browser for most tasks must be due in part to the relatively low complexity of the structures. In some hypertexts, displaying all nodes and links results in an incomprehensible cobweb on screen, forcing the user to filter – this was not the case with the structures used here. If, however, the full multiplicity of useful views were included within a QOC (oriented to needs of users, implementors, the organisation, marketing, management, etc.), the potential for information overload would be enormous. Visual information management issues need empirical study, but in the context of using realistic information bases, media, and tasks. Examples of relevant work on information presentation and navigation include Fairchild et al (1988), Shum (1990), Card et al (1991) and Robertson et al (1991).

3.2.8 USE OF CRITERION TREES IN THIS STUDY

Criterion trees were devised in order (i) to provide additional information to QOC graphs (inter-Criterion relationships), and (ii) to explore user problems integrating information between different representations. This section describes the use of Criterion trees in this study. Although the *form* in which incidents arose was to some extent a function of the paper-based media used (i.e. subjects had to perform tedious 'system' tasks which were unrealistic), incidents also point to software tool requirements relevant to integrating Criterion trees with QOC.

Most subjects were able to appreciate quite quickly what Criterion trees represented, how they were organised, and had few problems in knowing when to use them. All subjects were able to use the 'bold Criteria' to assess a system's 'Criterion profile' – which Criteria supported decisions within that system.

One of the purposes of an idea processing tool is to shift some of the representational load from the user to the system – to externalise cognitive representations. In §3.2.5.2, incidents were described where failure to use Criterion trees for certain queries resulted in subjects trying to construct mental representations functionally equivalent to parts of the Criterion tree in order to solve the query. Given that the other subjects used Criterion trees effectively, this strongly suggests that they reify important (task-related) internal representations – subjects constructed their own versions when they thought they were not available as browsers.

However, little has been gained if the Criterion tree is only useful for a set of unrealistic experimental tasks. Whilst some of the queries were designed to require information which only the Criterion tree could provide, this circularity is only problematic if the queries can be shown to be unrealistic; if the queries are judged to be valid, then Criterion trees can be held as being useful representations.

What is a 'realistic query'? This is where an element of uncertainty arises. In simulating use of an approach to design which has yet to gain general acceptance within the design community at large, it is impossible to develop queries with complete confidence that they are representative. All of the queries used in this study were developed assuming the widespread use of QOC-style representations and powerful hypertext tool support. Within this scenario, the claim is that it is reasonable to expect a user interface-oriented QOCs to answer queries about, for instance:

- how common operations are performed in the two systems (e.g. queries 9+11),
- why a particular feature is not present (7),
- the impact of architectural constraints (14),
- the use of psychological criteria (10),
- decisions made for the same reasons (15).

The introduction [§3.1.2] showed that the sorts of queries which other DR researchers anticipate as useful and realistic are similar to the ones used in this study.

Another source of problems encountered by subjects was in mapping Criteria from the Criterion tree to the QOC graph (S7 and S1). S7 felt that ideally, he should be able to click on a Criterion in the Criterion tree and see it highlighted wherever it occurred on the graph. S2 observed that bridging Criteria on the Criterion tree which mentioned the *Option* or *Decision* to which they related, were located much more quickly, e.g. "simple conceptual *link* model" and "no need to treat *buttons* as text", where the italicised words (emphasis added) provide spatial and semantic cues as to where the Criteria might be located in the QOC. S1 made the same observation, and also suggested that when mapping *from* QOC *to* the Criterion tree, the most helpful Criteria in the QOC were those which gave a clue as to the general 'type' of Criterion they were, e.g. ones which mentioned *tasks* cued subjects that these might be found under a parent Criterion to do with supporting task requirements.

Rather than incorporate 'clues' in the names of Criteria, as suggested by subjects, in a software tool one might instead expect to access an 'attribute sheet' which listed as hypertext nodes the Questions, Options and Decisions to which Criteria related, i.e. its local context of use within the QOC. As S7 requested, one would also expect that these links would be invisibly maintained until needed. An alternative to highlighting Criteria on a QOC would be to display a matrix of Options against that Criterion for each Question it related to—clearly, once implemented, additional improvements would suggest themselves.

An additional requirement is that the tool should be able to compute new virtual representations given a structural description by the user; these would be explicit bridges between the two representations, for instance, in order to explore the implications of introducing a new metaphor, it might be deemed useful to find out where the conceptual model in the current software release had been a factor, e.g.

Find all Options dated 1/1/90 onwards which were assessed by Criteria classed under conceptual model, and display with Questions and Criterion

The above examples of node-naming exemplify the more general problem of summarising ideas in a brief title which will be understood later on, possibly by other people. As described in Study 3 [§6.2.3], naming and renaming entities is one of the main overheads in authoring intelligible DR.

3.3 General discussion: Taking DR retrieval seriously

Having presented and discussed the results of this study, the key issue, namely, the *accessibility* or *coherence* of DR, is considered further. The potential of Criterion trees for supporting both DR retrieval and reuse is considered, before conclusions are drawn on the importance of coherent organisation to DRs.

3.3.1 DEVELOPING CRITERION TREES TO REVEAL BIASES IN REASONING

The use of Criterion trees in this study was described in the last section. The following analysis explores the potential for extending Criterion trees to more effectively support access to the DR, by providing a different perspective.

Let us begin by considering a criticism of Criterion trees by one of the subjects. S7 argued that from his own knowledge of the work of Xerox designers, it was likely that despite what the NoteCards Criterion tree said, most of the Criteria represented *would* have shaped the design of NoteCards, that is, he believed that most of the Criteria on the tree should have been bold. It followed from this, that differences between HyperCard and NoteCards – as claimed by the Criterion trees – were not credible. S7's comments about the Criterion trees strike at some key issues for the development of the Criterion tree concept, which are now drawn out and considered in more detail.

It was noted in the introduction to Criterion trees that a decision was taken to produce relatively simple paper-based trees, without much of the information which could potentially be added. A primary role which Criterion trees were envisaged playing, was as a means of understanding *differences in design*, through the *emphases in design reasoning* towards different classes of Criteria. As they stand, Criterion trees provide the following information:

- the hierarchy shows the more general Criteria which a bridging Criterion serves;
- Criteria which support Decisions (as opposed to rejected Options) are highlighted in bold (the status of other Criteria cannot be deduced: they could support or object to Options, or object to Decisions);
- the Criterion trees can be directly compared, because they have the same structure – this facilitates comparisons between trees.

For the task used in this study, Criterion trees in this form proved to be a useful representation, although even within these constraints expressive limitations were exposed [§3.2.8], and the limitations of paper-based media became evident. Clearly, in order to be of use in the wider context, Criterion trees must be enriched informationally, and possibly restructured. The following sections focus on a number of properties of QOC structures which Criterion trees must be able to take account of:

❑ the QOC's *coverage* of the design space;
❑ the *expressive form of Criteria*;
❑ the *size* of the QOCs being compared;
❑ different *frequencies* with which Criteria are used in a QOC;
❑ different *weighting* of Criteria, and importance of decisions.

These factors are discussed below in order to clarify how the representation is extensible beyond the constraints of the present study, for comparing the greater variety of DRs which would occur between different designers on different projects.

3.3.1.1 Qualitative and quantitative coverage of the design space

This is the main factor responsible for S7's critique of Criterion trees. Out of necessity, the QOCs only covered a small portion of the design space for each system. Therefore 'good' Criteria which one would expect to have supported decisions, and probably did in the actual system, were not bold on the Criterion tree, because those decisions were not represented – the Criterion tree only represents Criteria *used in the QOC*. As it is impossible for a QOC to cover every decision, this limitation was particularly aggravated by the use of such small representations.

The solution may not be found simply through larger QOCs; the *qualitative* element to the QOC is a key factor – *what kinds of Questions* are asked? Whilst there is no such thing as a 'complete QOC' since decisions can be represented at infinite levels of detail, if it is ensured that the reasoning is available underlying *key decisions* which shape the design, then the resulting Criterion tree presents a succinct overview of the design's 'philosophy' – the goals and priorities used in decision making.

3.3.1.2 Expressive form of Criteria

The Criterion trees and QOCs used in this study were the first representations developed in this research project, before QOC form had been analysed to the extent that it now has. A number of 'criteria for Criteria' emerged from subsequent studies [§10.3], which several of the Criteria here failed to satisfy. Criteria were unfocussed, that is, they embodied more than one Criterion, others were not positively phrased as goals, and others asserted *claims*-like relationships between Options and Criteria (Carroll and Kellogg, 1989), rather than remaining independent of specific Options. Some examples are shown in Table 3.9, together with the appropriate principle for expressing Criteria, and a suggested improvement.

Criteria in QOCs & on Criterion trees	***principle violated***	**alternative 'well-formed' Criteria**
high CPU load	*Criteria should be expressed positively for consistency and intelligibility*	low CPU load
abstract actions difficult to iconise	*Criteria should be expressed positively for consistency and intelligibility* *Criteria should remain independent of specific Options*	clear icon semantics
only useful to display different links graphically on a *system supported* browser [an earlier decision had rejected use of a browser]	*Criteria should only articulate goals; dependencies between decisions should be represented as separate entities*	[once QOC is implemented within a tool, dependency could be represented as] a *claim* that there is an inconsistency with an earlier decision (also called *imports* by MacLean et al, 1991) or, a *dependency link* to the earlier decision
sophisticated editor possible for each card type	*Criteria should remain independent of specific Options*	high editor functionality
use clear informative screen design	*Criteria should embody only one goal of interest*	[if clarity and information-content assess Options differently use separate Criteria:] clear screen design informative screen design

Table 3.9: Examples of Criteria with poor 'expressive form' as used in the QOCs and Criterion trees in Studies 1 and 2.

It is of interest to note that errors of this sort were replicated in the subsequent studies of QOC authoring, particularly Study 3 [§6.2.3]. As such, it would appear to be a robust phenomenon, characteristic of early use of QOC, and presumably other DR notations.

3.3.1.3 Handling different sizes of QOC

Since a larger QOC will invariably use more Criteria, it will have a larger Criterion tree, which renders meaningless simple comparisons between trees of the number of Criteria in a class. To compare how two designs were influenced by a class of Criteria, bold Criteria in that class could be expressed as a proportion of all *decisions*. Thus, if a QOC contains 20 decisions, five of which are supported by Criteria of class X, the Criterion tree could indicate that 25% of decisions made satisfied Criteria of class X.

3.3.1.4 Handling reused Criteria

This is a second complicating factor: it is possible for one system to have many bold Criteria in a class, but used infrequently in making decisions, whilst another has fewer bold Criteria, but used many times. The solution is again to use a proportion, expressing the frequency with which a bold Criterion appears as a *proportion of all bold Criteria*. Thus, if there are 20 bold Criteria in total in a QOC, and a particular class of Criteria on the tree has five bold Criteria, each of which is used twice, each bold Criterion in that class accounts for 10% of all bold Criteria and the class accounts for 50%.

As a Criterion is never used twice within a given Question, every occurrence of a bold Criterion represents another decision supported. To calculate the extent to which a *class* with reused Criteria supports the design, the tool could simply count the number of decisions in the QOC supported by one or more of the Criteria in that class, and express it as a proportion of the total number of decisions.

3.3.1.5 Weighting Criteria

Inevitably, there will be decisions in the QOC which are considered more important than the others. If Questions had an attribute of *importance*, then it would be a simple matter for the Criteria assessing the Options to key Questions to be automatically marked on the Criterion tree.

It is inevitable that some Criteria will be prioritised above others, from the start of the project – decisions are not made solely on the basis of the *number* of Criteria supporting and objecting to Options. The need to express weighting of Criteria and assessments was a prime concern of designers in Study 3 [§6.3.3 §6.3.4], as well as in Case Studies 1 and 2 [§7.3; §8.2.1]. As a Criterion tree summarises all the goals of a design used in making decisions in the QOC, it is an obvious representation via which Criterion weightings could be assigned and modified.

In brief, by adding power to the representation in the ways outlined above, it becomes increasingly possible to compare different QOCs meaningfully via their respective Criterion trees. Note also, that although the Criterion tree becomes more complex as it is made more powerful, extending it in the ways suggested involves no extra cognitive effort on the part of the QOC author. The more expressive the QOC is, the more powerful its Criterion tree can be, since Criterion-reuse, significance, and QOC size are attributes of, and hence computed from the *QOC* – it is in constructing QOC where the effort is invested.

3.3.2 CRITERION TREES AS REUSABLE STRUCTURES

DRs produced both within and between organisations will inevitably vary in the way in which Questions are used to structure them. This is the case even between DRs for the same problem, as demonstrated by the variation in QOCs created in Study 3. Different teams have different definitions and concerns. An intermediary representation which expresses a range of perspectives within a common framework could therefore be extremely useful.

The Criterion tree could be viewed as a step towards ‘normalising’ different QOCs. By grouping Criteria together under generic classifications at different levels of detail, the hierarchy is a way of assessing the use of Criteria in different QOCs. Clearly, a classificational scheme would need to be widely agreed as a standard; human factors guidelines (e.g. Smith and Mosier, 1982) provide a possible starting point for an organisational scheme for defining the hierarchy’s classes.

Just as Fischer et al (1991) envisage the use of *generic issue bases* [§3.1.2] in well understood domains, once *generic Criterion trees* were established, they would need only minor tailoring in order to incorporate classes of Criteria specific to a particular design. To a small extent, this is exemplified in the QOCs used in this study. The QOC for each of the two systems was organised around three main topics (represented as QOC subtrees). Virtually identical Questions were asked about the system model and user interface for *nodes*, for *links*, and for *navigation*. It would be to the advantage of the DR community at large if DRs for systems of the same sort could be viewed from one, or perhaps a set, of agreed perspectives.

One view might be based around the system image which a class of system commonly presents (nodes and links for hypertext; objects, layers, and users for collaborative drawing tools), whilst another view could be the generic tasks which users of a class of system expect to perform, for instance, *navigating*, *viewing structure*, and *modifying structure* for idea processing aids; *editing objects*, *sharing views*, and *being aware of others’ activity* for shared drawing tools.[6] Carroll and Rosson (1991) suggest that their work on psychological DR by claims-analysis could produce a typology of generic user concerns across domains. Clearly, the existence of such schemes reduces the overheads for the DR author; ultimately, authoring DR on a project could become a case of tailoring an existing DR, and the availability of sets of Questions would act as an initial guide for ideas.

6 See Bellotti et al (1991) for examples of QOC analyses of collaborative tools.

To recap, it is suggested that a Criterion tree, or an analogous representation (e.g. a goal hierarchy), is a good candidate as a means of integrating many of the Criteria to be used into a single, manageable representation. A possible scenario for use might thus be as follows: the designers identify requirements and any initial constraints (such as functional requirements and resource constraints which preclude certain Options). If known, dominant principles which will guide decisions are also made explicit (e.g. a particular system metaphor). The designers then consider a range of Criterion tree structures which have been found useful in the past for problems of this sort. The aim here is to both stimulate further ideas about relevant Criteria, and to tailor a tree, or perhaps combine several until they have a structure which provides a coherent organisation for the goals, principles, and desirable characteristics which have been identified. The Criterion tree organises them into a hierarchy as far as possible, allowing for cross-referencing of a Criterion under several classes as appropriate. As the project progresses, new classes which suggest themselves are added.

It is difficult to imagine how Criterion trees might be used in the future, and the above scenario is only one possible approach. Whilst proposals such as this might be considered speculative or premature (the basic concept of representing DR has yet to be accepted on a significant scale), there is no doubting the need to organise Criteria. Indeed, it adds credibility to DR if it can be demonstrated that the essence of DR – answering *why* – is grounded in more than the whim of designers; Criterion trees serve to promote internal consistency in design reasoning which is directed towards fulfilling the project's goals.

3.3.3 THE IMPORTANCE OF COHERENCE IN QOC STRUCTURE

It is clear from the results of this study that the coherence of the organisation of the DR is critical if it is to be accessible and reusable. Since the original paper presenting QOC as a representation for DR (MacLean et al, 1989) the characteristics of the Design Space Analysis approach have been clarified and extended considerably. One result has been that instead of a large graph with many consequent Questions (as used in this study), a QOC analysis takes the distinctive form of a set of Questions, elaborating different parts of the design space, connected by logical dependencies and constraints between Options which span the design space. One of the problems with the large QOC browsers was that subjects could not always follow the logic in a series of 'consequent' Questions. For example, S5 expressed surprise that the Question *how should cards be viewed?* should be "under" *how should a node be conceptualised?*

The following sequence of Questions and Decisions illustrates the problem:

> *Q: how should a node be conceptualised?*
> D: card
> *Q: how should a new card be created?*
> D: menu option
> *Q: how many card types needed for different media?*
> D: one type for each medium

By linking Questions in this way, the issue addressed by a Question is *embedded in a particular context*, which whilst conveying the order in which issues were considered, in so doing also reduces the accessibility of the QOC—this sequence may not match the context in which the QOC needs to retrieved subsequently. The extract above is not an *illogical* sequence, but to another designer trying, for instance, to find out why NoteCards uses different card types, it is not a sequence which necessarily makes sense or could be inferred. It is quite possible to imagine several different contexts (i.e. one could formulate alternative Questions) in which the decision might have been made, and it is unreasonable to require that designers wishing to check the DR should double-guess the original sequence of deliberations in order to locate the relevant information.

This is however precisely the situation in which users of a narrative DR, such as gIBIS, find themselves. In a gIBIS structure, the whole point is that the original order of deliberations is preserved. Issues are generally posted as and when they occur. As design problem solving and discussion is not a logical, orderly mode of working, without rationalisation, the resultant DR is poorly structured and less accessible to outsiders. In fact, Conklin and Yakemovic (1991) state very clearly that process-oriented DR is *not intended to support reuse*:

> In the structure-oriented approach, the DR is prescriptive, since it summarises the design decisions and trade-offs so that others will reuse the reasoning. In the process-oriented approach, the DR is merely descriptive; its reusability is incidental.

Since one of the key roles which DR can play is in providing insight into a system's design to others, this is surely a major weakness in the process-oriented approach. Recording DR now represents an investment whose payoff is relatively short-term (e.g. limited to the duration of the project).

There is at present no data on the reuse (or *use* for that matter) of realistically sized DRs, either in research environments, or the real world. An area which badly needs investigation is how useful the different kinds of DR are to different potential DR user communities, like system maintainers, or software engineers upgrading systems—what rationale do they use at present? The implications for DR approaches are clear: if designers seeking to understand a system are interested in what Conklin and Yakemovic refer to as "a dimension of narrative...[with] the wrong turns and rejected alternatives...[in] the order in which they were taken", then DR which includes historical process will be valuable. If they are more likely to come to DR with queries of the sort anticipated in this study, then retrospective, rationalised DR will meet their needs better, offering a concise presentation of the key issues and arguments behind the *current* state of the design. Structure-oriented DR is more formalised and rationalised at both the *macro-level* (e.g. where will DR on different kinds of design decision be found?; how are Criteria organised?) and at the *micro-level* (the extent to which DR is 'well-structured' at the level of individual Questions, e.g. how focussed the Questions and Criteria are[7]).

At the start of this chapter [§3.1.2], other work on DR retrieval were surveyed. It is telling that although there was a wide range of queries, virtually none required *narrative* rationale—that is, those aspects of process-oriented DR which distinguish it from more rationalised, structure-oriented DR.

It is perhaps possible that the literal history of a design meeting could be retrieved from retrospective DR, if it was constructed during that meeting or from recordings of it. If nodes and links carried an attribute of *creation-date*, then it is conceivable that the tool could compile a chronological list of entities. This would however depend on a versioning system of some sort, such that subsequent rationalisations of the DR did not overwrite the original trace. One can imagine the tool 'playing back' such a trace to reconstruct the DR's development, at some granularity of update. This may however be a crucial point: the memory requirements to save *every* change made to the representation would be enormous; if a larger granularity of storage was used, what should it be? These are open research questions which require implementations robust enough to withstand realistic evaluation.

In conclusion, to achieve its full potential, DR should be created to be retrieved and reused; it must therefore be rationalised retrospectively in order to effectively communicate the logical content and structure of the reasoning behind the design. This will inevitably result in at least the partial loss of the narrative structure which shapes non-rationalised DR, but it is hypothesised that this will not seriously affect the DR's usefulness, firstly because narrative structure is not of prime interest to the design communities who most need to reuse DR, and secondly, because it may still be possible to extract narrative content from a retrospective DR. It is proposed that the benefits of rationalisation for designers and 'reusers' of the DR, outweigh the benefits of leaving the DR in its original form.

The cognitive task of authoring coherent, reusable QOC DR during design problem solving, is investigated in the following studies.

[7] QOC form at this level is exemplified and discussed in the authoring studies [e.g. §4.2.6.2, §10.3]

Chapter 4

Study 2: Creating, Reusing, and Integrating QOC DR

4.1 Introduction and experimental methodology

Study 2 was a continuation of Study 1. Having been placed in the role of a designer seeking to understand two hypertext systems, subjects now moved into a redesign phase, in which they were asked to modify one of the systems, integrating the rationale for their designs with the existing DR. This study was thus conducted as an initial investigation into issues relating to QOC authoring, the reuse of existing QOC for redesign, and the integration of new QOC with old. As subjects had received no training or practice in authoring QOC, the purpose of these 'minimalist' conditions (in terms of the subjects' expertise, and the representational technology used) was to force to the surface problems which could be studied subsequently in more detail.

4.1.1 Subjects

Study 1 subjects were reused in this study (see §3.1.3.3 for details). They were therefore practised to some extent in translating information requirements into QOC, and in visually scanning QOC graphs and lists. However, they had not up to this point expressed their own ideas as QOC.

4.1.2 Materials

Materials were as in Study 1: subjects used pen and paper to represent QOC and their designs, and QOC browsers and Criterion trees were available on request to the experimenter, at all levels of detail in both graphical and list format.

4.1.3 Design task and conditions

The design task presented a scenario in which subjects continued in their Study 1 role as NoteCards system maintainers, but now wished to add a new operation to create *composite nodes*. An example was provided (Figure 4.1), to illustrate the pulling together of a set of nodes into one graphical object, the composite node. From this example, subjects had to design a facility with this functionality.

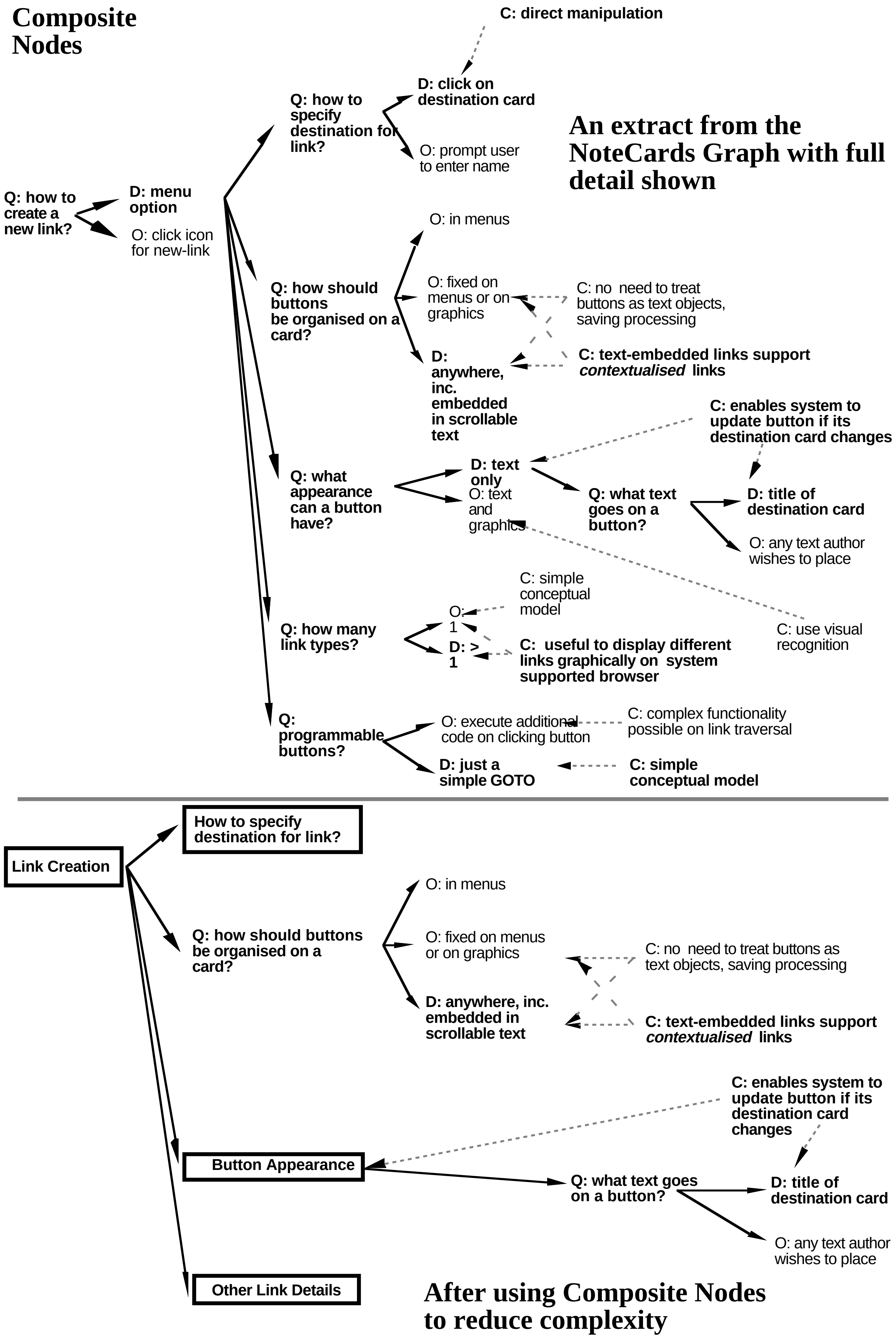


Figure 4.1: Example used in the Study 2 design task (Composite Node facility task)

Subjects were allowed to ask the experimenter questions about the NoteCards interface if they could not find the information in the problem statement or DRs. They were instructed to try and maintain consistency with the existing interface, and to justify their design in terms of the general Criteria used in the design so far, i.e. decisions should as far as possible be compatible with existing DR.

As a further exploration of potentially relevant factors, a variable was manipulated which the available literature suggested was important, namely, the point at which QOC was used during the design task. The analysis of writing cognition and tools, together with growing evidence in the hypertext literature of user problems in segmenting ideas into discrete nodes focussed attention on this usability issue in particular [§1.3.2]. Consequently, a late condition was added to the design task. S6 and S7 only used QOC *after* they had spent the first half of the session (30 minutes) working on the problem. It was hypothesised that subjects would experience more difficulty using QOC concurrently during idea generation, rather than retrospectively to organise ideas which had been worked through to a greater extent.

Thus, S1-5 were instructed:

> Create a small design list or graph, showing the design decision history which might underlie the new *create composite node* operation.

This will be referred to as the *concurrent DR* condition. In the *retrospective DR* condition, S6 and S7 were told,

> The first job therefore is to design this new feature. After you've designed some of the necessary features, I'll ask you to organise those decisions as a design rationale (graph or list as you prefer). However, concentrate on the design task initially.

4.1.4 PROCEDURE

After an optional break following the Study 1 task, subjects commenced Study 2. They were allowed one hour to tackle the Composite Node task, once the instructions were understood. As in Study 1, the session was recorded on videotape and the experimenter interacted with subjects throughout the session.

4.2 Results and discussion

The results from this exploratory study characterise very early experience in QOC authoring. Observations are grouped under several main headings, some task related (integrating new ideas into existing DR; managing dependencies), others based around QOC (using Options; using Criteria). For each section, the general issue is introduced, and relevant data presented and then discussed, rather than leaving all general discussion to the end. Because of the scope of the study, in many cases there are only a few instances of a class of behaviour. These are used however as a springboard for ideas about more general issues, referencing where appropriate additional evidence or insights gained subsequently through Study 3 or the Case Studies.

4.2.1 ADDING NEW DR: PLACING IDEAS IN CONTEXT

One of the first problems faced by the designer wishing to add a new idea is to decide where it best fits in with existing understanding of the problem, specifically, with the QOC as it currently stands. Initially, poor understanding of the problem may limit expressions to vague statements about the general topic to which the idea relates, for example, comments by several subjects that an idea is "to do with navigation" or "relates to the problem of link-types."

At some point, however, authors must move from this underspecified notion of the relevance of an idea, and work out how to express it in QOC. *Is it a new Option? – to which Question?; is a new Question required, or should an existing one be modified?; what Criteria can be used to justify it?* The design task required subjects to integrate their new ideas into the existing QOC, which led to several different QOC structures. The issue was the extent to which interpretations of 'composite nodes' as a concept would vary – how many different ways were there to represent the same idea?

Before considering the different ways in which subjects integrated new DR with old, it is interesting to note that all subjects chose to use the graphical browser to work out how to link in the new QOC, confirming general opinion in Study 1 that lists by comparison offered a less complete view of the information. Whilst the *full detail* graph was employed by all subjects, a minor elaboration on this strategy was added by S7, who used the most filtered graph, which revealed the basic three subtrees most clearly, to decide the *local* space for linking-in, and then switched to the full browser to get as much contextual information around *graphical browsers* (the Decision of interest to him).

Attention now turns to the specific DRs produced. Perhaps the simplest way in which a new feature can be added to a QOC is to locate (if possible) the Decision to implement the functionality which is being modified (i.e. graphical browser), and ask a consequent Yes/No Question as to whether to add the new feature (composite nodes). S2 viewed composite nodes as a feature of *graphical browsers*, adding a Yes/No Question *provide composite node facility?* to the Decision *graphical browsers* in the existing DR. Figure 4.2 shows how S2 constructed his DR.

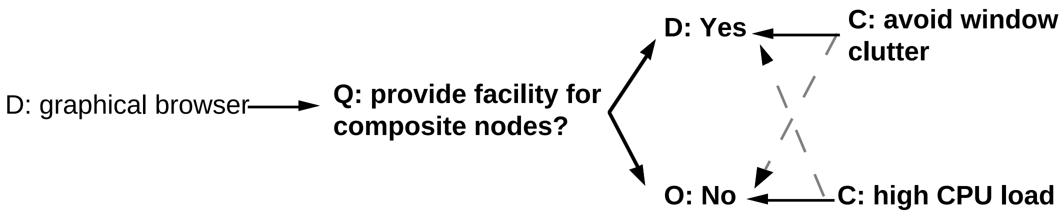


Figure 4.2: Linking in new DR by attaching a consequent Question to an existing Decision (new QOC shown in bold)

Whilst being the simplest approach, this is not a particularly useful representational strategy for expanding the design space. The use of Yes/No Questions about specific Options 'violates' a principle for formulating Questions, reviewed later [§4.2.6.2, §10.3]. A better approach might be to consider what function composite nodes serve, possible answers to which might be *reduce clutter*, or *collapsing a subtree into a composite could indicate its status*. From these ideas, one might then formulate a Question, such as *how to indicate status of a subtree?*, or *how to reduce browser clutter?* If it was decided that *subtree status* was the key issue, but that *minimising browser clutter* was also important, it could still be incorporated in the analysis—as a Criterion to evaluate the Options.

Lack of knowledge about hypertext caused a number of problems for S3, S5 and S7 when it came to integrating new QOC with old. S3 was able to reuse psychological Criteria from the Criterion tree to back up Decisions, but when asked to integrate the new QOC with existing structure found it impossible. This was attributed by the subject to a lack of domain knowledge about hypertext in general, and about NoteCards in particular. S5 also felt that he was not sufficiently familiar with NoteCards.

Looking at the Criterion tree, S5 decided that composite nodes could be linked-in under *navigation*, as they show the network more clearly helping navigation. He was able to articulate quite specifically the sort of Question which would be the ideal point to link in his DR (*Q: what is conceptual model for a link?*) but there was no such Question. S5 concluded that the existing QOC needed to be modified to achieve a better logical structure (subjects were allowed to make changes to any part of the existing DR). Interestingly, this subject wondered if editing the DR equated to re-writing design history.[1] The modified structure is shown in Figure 4.3.

1 This concept of what QOC represents (a true record of the design process) was probably partly due to the fact that he was using QOC concurrently to record his own design.

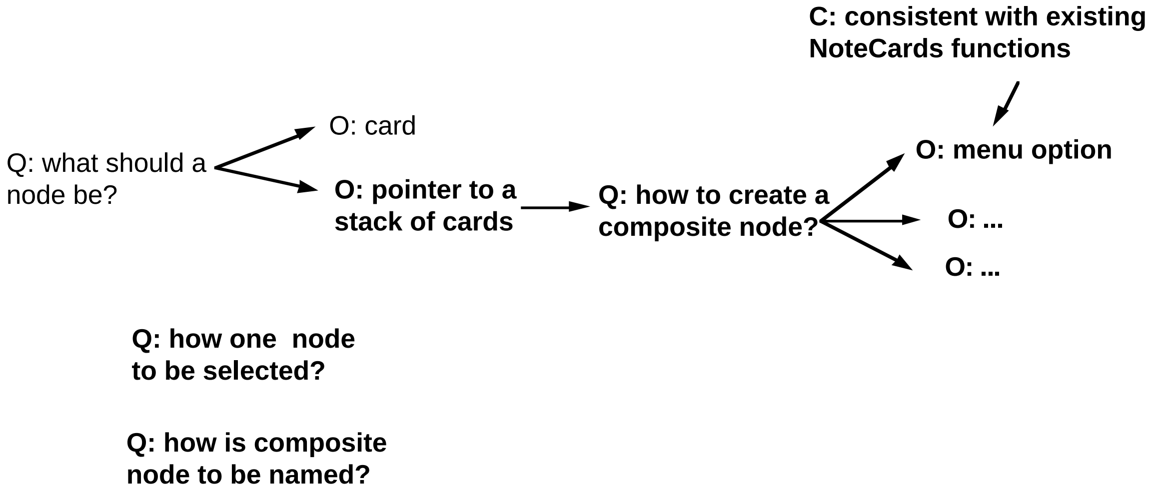


Figure 4.3: Conceptualising the composite node as a new node type: composite node is added as an Option to an existing Question about nodes, and a consequent Question explores that Option further (new QOC shown in bold).

The task of linking new QOC into old proved problematic for S4. As a computer scientist familiar with software design representations in his own discipline, maintaining formal consistency was an important consideration. He expressed some concern over QOC's lack of support for this task:

> You see, we've really got two issues here: not only do we have the design issue, but whether we think this (QOC) structure is a good structure in itself. As a designer, I wouldn't expect to be asking these questions – how do you introduce a new element into the design? ... if you follow SSADM you have a particular path of refinement, and in databases you have 'rules of normal form' which allow you to almost mechanically do these sorts of operations, and give you some notion that what you do is going to produce a consistent representation.

Concern for consistency led S4 to think about reusing the structure of the existing QOC in some way. One possibility was simply to add a new Question subtree devoted to composite nodes, and ensure that relationships to the rest of the QOC were added (such as consistencies, inconsistencies, and dependencies).

However, another way was to add extra Options and Criteria to corresponding parts of the existing QOC. He posed a Question on reducing clutter (setting composite nodes in context) and then reused the existing Questions on nodes, links and navigation as a means of integrating the new rationale for composite nodes. Figure 4.4 shows S4's QOC, augmented to reflect his description of how he was using it (Appendix 2 reproduces the original form, which was incomplete).

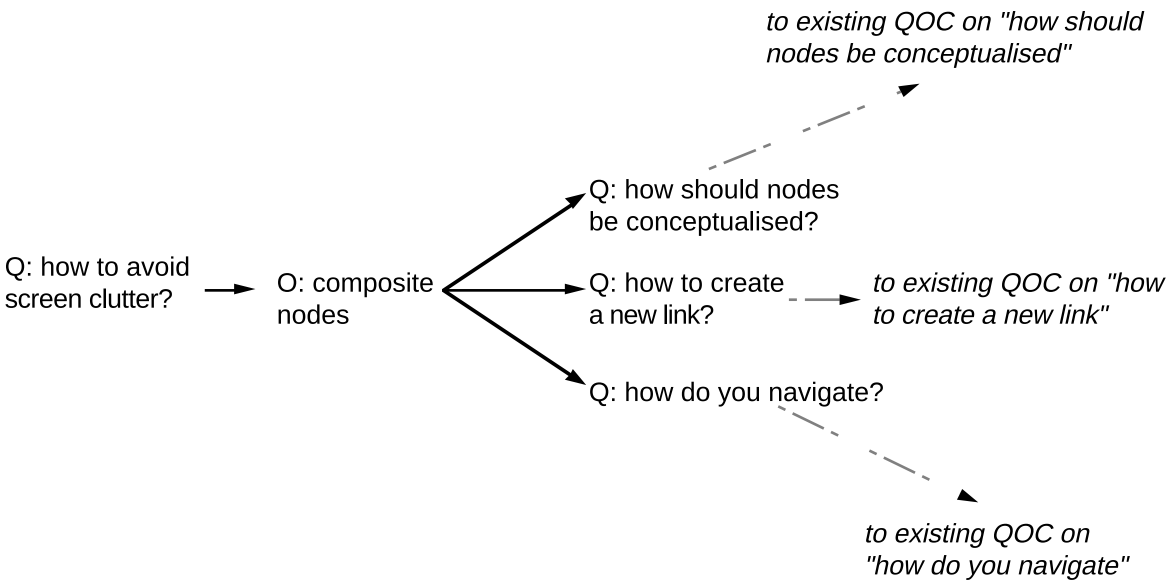


Figure 4.4: Reusing the Questions in the existing DR to link in new DR for the composite node (redrawn from S4–original in Appendix 2).

He pointed out that this followed general practice with other design methodologies in which one tries to reuse components in order to maintain consistency:

> From the composite nodes, the obvious Questions to ask are *how should a node be conceptualised, how links should be created* and *how to navigate*, because those are the Questions we've addressed already ... is a composite node a special sort of link? – in which case [existing QOC on] the creation of a new link has to be refined into specifications of how to create a normal link and how to create a composite node link.

In other words, the introduction of composite nodes has implications for each of the three main Questions in the existing QOC, for instance:

the concept of a node:
❑ does the interface metaphor for a node need changing?
❑ can a node occur in more than one composite node?
link-creation:
❑ is the existing link-model adequate for composite links?
❑ if a composite node is linked to something, are all of its contents?
navigation:
❑ how to get to a node if 'held' within a composite?
❑ do existing navigational methods apply to composite nodes?

The reuse of QOC structure in this way serves as a checklist of points against which the designer can assess the implications of new ideas for the rest of the design.

4.2.1.1 Adding new DR: Discussion

The above examples clearly point to the representational variation which QOC (and other DR notations) permit. Two factors contribute to this: (i) the *semi-formality* of the notation means that a given idea can vary in the way it is expressed between authors; (ii) the lack of any *organising principles* by which new ideas should be translated and integrated into QOC leaves the decision wholly up to the designer.

Does it actually matter that this level of informality exists? As reported, one subject viewed the lack of any guiding principles in organising QOC as a weakness in the approach. He was effectively arguing that the *absence of constraints* in fact makes the task of authoring QOC harder; certainly, Study 1 [§3.3.3], showed how poorly structured QOC created problems for retrieving information.

The answer surely depends on the role which the QOC is playing. If the QOC is acting as a relatively *small* working representation for the *immediate design* team, then variations in the way in which QOC is recorded are less important. There will be few situations in which different designers are representing the same decisions as QOC, and minor differences will not matter. However, if others need to understand the QOC and subsequently add to it, then there must be some coherence to the macro-organisation, or retrieving and, as has now been demonstrated, *integrating* rationale, is seriously complicated.

4.2.2 Summary of strategies in using QOC

This section describes the different *processes* through which QOC structures were created. The following observations cover the use of QOC in both conditions – *concurrently* with design problem solving, and *retrospectively* after some problem solving had been done. Different strategies and patterns for translating ideas into QOC are also documented in Study 3 [§6.2.1].

S1 started by thinking of Options explicitly, then worked out the Questions, ignoring Criteria. For this subject, the Options came more clearly than the Questions initially. S2 described his approach as follows (the strategy adopted by all subjects):

> *What determines the order of the Questions you're asking?*
>
> I suppose it's a sort of implicit analysis of the information the system is going to need in order to carry out the task. What I've done is come up with a first pass design in my head, and then linearise that. So I've thought that a sensible way of doing it would be to first of all say what nodes I want to collapse (into the composite node) and then think about how might one do that—what are the consequences, do we need to worry about non-hierarchical structures and exception conditions. And then we have to give it a name, and presumably the system will do the rest and redraw the screen.

S3 started by jotting down four key questions which would need to be answered. She did this by mentally 'walking through' the task (as above), commenting that more Questions might have to be added later. This in fact happened, when one of the four Questions posed as *where is unfolded section displayed and how?* was later broken down into separate *where* and *how* Questions.

As one of the subjects who spent the first half of the session working without QOC, S6 had given the design some thought before beginning on QOC. Whilst S7 showed no obvious benefit from having a problem solving period before using QOC, it was noticeable that S6 was able to record the Options to the first Question rapidly, and later when he noticed that some ideas he had discussed earlier were missing from the QOC, he quickly added two more Questions plus Options, without pausing for long periods to 'find' other Options.

However, it became clear that S6 had not worked out the *reasons* for his Decisions, and he was forced to slow down as he tried to devise different Criteria which supported and objected to his Decision. In so doing, he commented that a Criterion objecting to his Decision served to raise another Question:

> Negative Criteria generally just suggest Questions ... putting in a negative Criterion is just another Question about it. I was going to put in a negative Criterion saying *very hard to unpack* because as I envisage the design so far, I can't think of a way of unpacking a composite node in the way I've just described it. But that's also a very good Question which I need to go on and expand.

This led to a new Question on extracting nodes from composites (but for some reason he did not add the objecting Criterion, as though it was redundant). A convention for 'well-structured' QOC [§10.3] is to record the Criterion, but then to try and overcome it by 'optimising the Decision' through a consequent Question (a heuristic proposed by MacLean et al, 1991). The fact that S6 effectively articulated this heuristic himself, suggests that at the very least, the heuristic is a natural translation into QOC language of a process in which designers clearly already engage.

Clearly, it would be desirable to be able to present evidence that DR notations facilitate design problem solving. A problem, however, which was encountered several times during this research, is that it is difficult to prove that an incident is due wholly or in part to use of the notation. Whilst in a small scale study such as this, it would be surprising to find benefits clearly attributable to the use of QOC, incidents such as the one above illustrate how QOC can be integrated into the problem solving process, and that it may even support it, for instance, through the application of heuristics such as 'overcome negative Criteria.'

Whilst using QOC may have helped in the last example, using it concurrently raised problems for other subjects. S5 commented that designing in the order which QOC suggests was "a very funny way to do design." His comments are reproduced below, as they characterise many of the problems encountered by subjects in this and subsequent studies:

> Fitting things in on this piece of paper in terms of Questions, Options and Criteria... it's kind of... you know, if I come up say, thinking – let's have a dialogue box which says 'that' – then I have to think about what is 'that'? What Question is that answering? It's not always the thing, that first you want to think about the Questions, then the Options... sometimes you want to work backwards.
>
> And sometimes you know that there are other things you want here (indicates different area of QOC) but you can't put your finger on. So you get the feeling that you ought to do all this first (all Options and Criteria for current Question) before you move on down to here (consequent Questions).
>
> It's not the way I think, and similarly, if I come up with a thought like I was doing just there, I have to think, "Is that a Criterion?"—it's a very important thing to consider, but now I'm being forced to think does that want to be a Criterion there, and is this Question actually in the right place there, or should it be back here? What difference does it make to the actual thinking process?

It is interesting to note that S5 felt constrained to designing within the notational order Q→O→C, even though there was no requirement as such in the instructions.[2]

S5 closed with following comment:

> But again, I guess one way to get around that is to say, let's not worry about representing it in this way – let's think about the design, and then record it after the fact in this way – maybe that would be a better way of doing it.

This mode of working is in accord with the retrospective approach to DR advocated by MacLean et al (1989).

4.2.3 MANAGING DEPENDENCY

This section focusses on one incident demonstrating the importance of being able to represent dependency in QOC. Whilst this example was the clearest of the incidents encountered, Study 3 describes further examples [§6.3.1.3].

As S6 considered the problem, and developed the QOC, it emerged that the path followed from an early Question had to be changed (this Decision had been made early in the session, before recording QOC). It was found that none of the Options for a later Question could be chosen without violating the main Criterion (avoiding modes) used to make the earlier Decision. S6's QOC is illustrated in Figure 4.5, together with annotations to show the conflict.

---

2 In training Case Study 2 designers, particular effort was made to emphasise the *non-linear* model of design which QOC assumes (in contrast with top-down models of the design process), so that designers should not feel constrained to having to open with an initial Question, or 'complete' one Question's Options and Criteria before moving to the next. However, those designers still felt constrained by the approach to following the sequential order implicit in the notation.

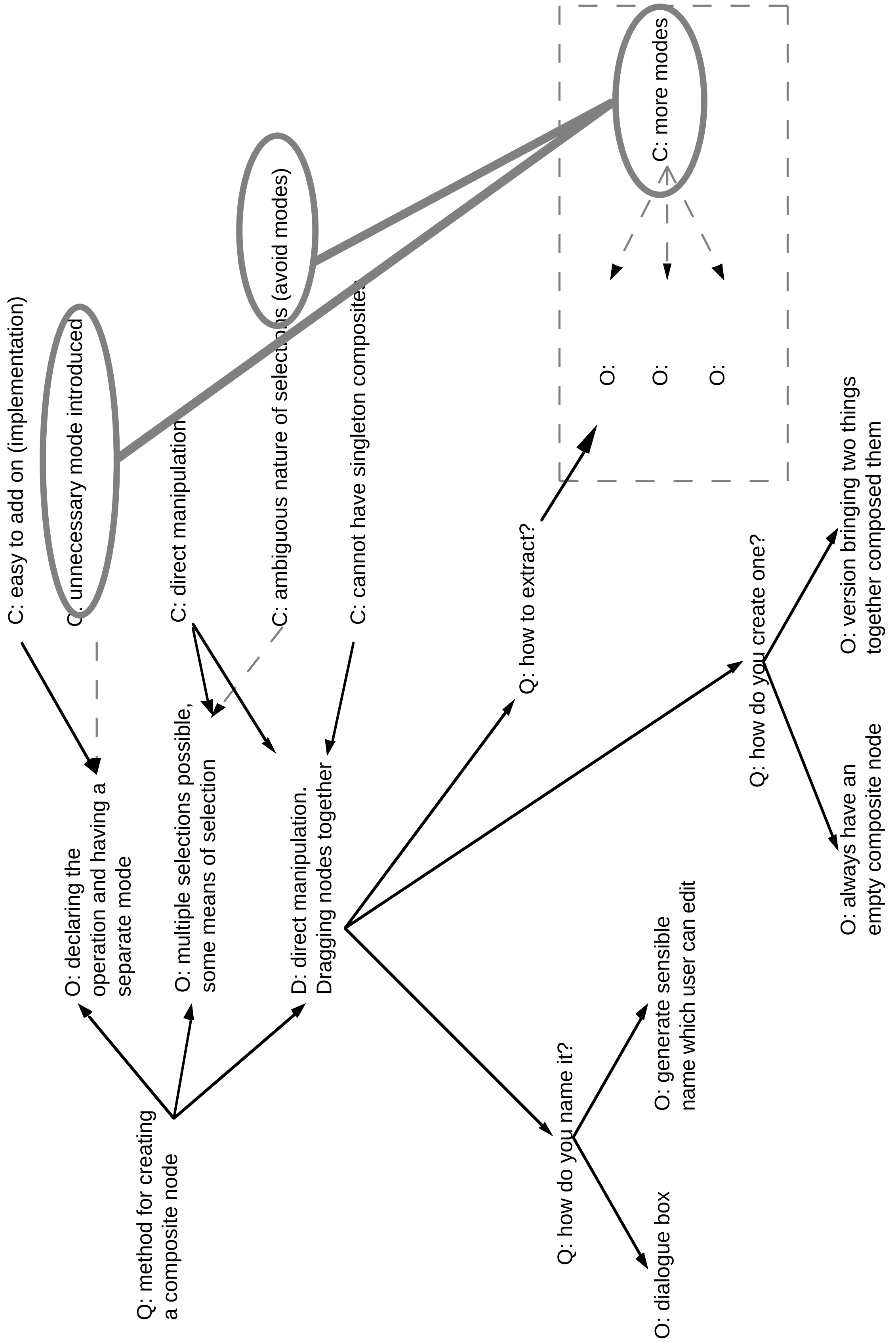


Figure 4.5: S6's QOC, annotated to show the inconsistency detected between early and late Decisions.

The later Question could not be resolved without introducing new modes, but earlier Decisions were made on the basis of Criteria which explicitly avoided modes. Given that modes are now inevitable, S6 wonders if he should backtrack and reconsider the path he has taken:

> I've got to adopt one of these (Options to Q: *how to extract?*) because I've got to extract - if I'm forced to adopt one of these...then these two (circled *modes* Criteria) don't hold anymore . So if I swallow my pride about these two (rejected Options to first Question) and say they *are* alternatives, then I've got to start opening up Questions about them, whereas I've already answered quite a few of the Questions down here.
>
> What I'm getting at is that I could ...have rejected these Options (to the first Question) 6 months ago, but now I find that I'm forced to adopt Decisions which negate my objecting Criteria 6 months ago, and I don't want to have to start doing all that again.

As S6 went on to comment:

> The design space – the actual thing they're trying to represent – changes, as you represent it ... everytime you make a Decision, three more appear somewhere else, but if you'd said yes to a different Decision, these three might never have come up. It's non-monotonic: the order in which you make Decisions matters, and changes everything which happens in the future. The design problem in general is that there are too many things to decide at one point, so people have all sorts of criteria about what you decide first, like first your hardware, then your programming language, and so forth.

The heart of the problem in this incident arises from the nature of design – commitments to a path always risk being subsequently proved incorrect as its implications are explored. The issue for QOC, is that if such paths are documented *during design*, the overhead in updating the structure must be minimised. Keeping track of a dynamic structure with complex interdependencies places an unreasonable load on the user, and use of QOC must pay for the additional representational overheads created. Because the structure was so small, S6 noticed the problem himself, but given a large structure it would be much more complex, and computational services such as SIBYL's dependency management [§2.1.6] would be needed.

4.2.4 USING OPTIONS

A fundamental tenet of DR is that it is useful to see the Options which were rejected, as well as the final Decision. This section reports findings in this study which point to the cognitive tasks that generating Options involves.

4.2.4.1 'Forcing' the Options into the open

One situation which arose on several occasions was that no alternative Options had actually been considered, but the subject felt 'under pressure' to find alternatives. On one occasion, S3 simply used extreme contrasts as Options due to lack of ideas (e.g. command-based vs. menu driven), behaviour which was observed in several other subjects. The feeling was that when they have made a Decision, designers do not want the trouble of having to think of alternatives. In one instance, S7 decided on the Decision to one Question before any alternative Options had been considered; he knew what he wanted to do, and did not want the overhead of devising possibly spurious alternative Options.

Nonetheless, there was also apparent benefit in pausing to generate Options. S7 reconsidered Options to a Decision he had made (to vary menu-length depending on the context), which led to an alternative solution which had not thus far been considered (greying out irrelevant menu items, maintaining consistent spatial layout). However, as was noted in an earlier example [§4.2.2] it is difficult to prove that this idea emerged through the use of QOC. It is reasonable to claim, however, that QOC's emphasis on laying out Options 'made space' in the deliberation process which allowed the better Option to suggest itself.

4.2.4.2 What counts as an Option?

In considering Options, the question was asked (S2) "should I record Options which I know are not feasible or which later emerge as not feasible?" (where "not feasible" meant "won't do the job"). Similarly, S5 wondered if an Option can be something which is simply considered during problem solving, or whether it has to be "really possible."

If the QOC is developed along with the design, this can cause particular problems. An Option may appear to be quite feasible, but later has to be rejected (e.g. on spotting hidden dependencies [§4.2.3]). There is little to be gained by being dogmatic about what is and is not a legitimate Option; if a design path seems promising, and is pursued only to find later on that it is not feasible, then representing that path could prevent others from making the same mistake. A negative Assessment might summarise the problem in a few words, but there could be an indication that more detailed argumentation was available if needed (e.g. by clicking on the negative Assessment).

Although with semi-formal notations, it is difficult to set rules as to what 'qualifies' as a given node type, nonetheless, there emerged from the combined data of all the studies a number of properties which a well-structured QOC displays. Whilst these are collated and expanded later [§10.3], the relevant representational conventions for Options are summarised below:

- Each Option has positive and negative Assessments
- Options are potential design Decisions
- There is one alternative per Option
- Options are generally distinctive alternatives, not minor variations on one Option (or), Options are expressed at a consistent level of detail within a given Question.

4.2.4.3 Representing the status of Options

The degree of confidence which a designer has in a design path cannot be expressed notationally at present. Clearly this issue applies primarily to concurrent use of QOC during design, when commitment to a Decision may bring with it unknown consequences. Subjects would consider the implications of Options in turn, often in their own minds rather than recording ideas on paper. When a Decision was made, the ‘O’ for Option was changed to a ‘D’ for Decision, and the consequent Question considered. If a tool rather than paper and pen were used, it might be useful for users to create ‘soft rationale’ from a tentative Decision, which could be quickly ‘undone’ if a line of reasoning dried up, deleting all QOC back up to its root Option, or ‘pasted’ to another Option for re-use. §10.7 summarises the implications from all of the studies for DR tools.

4.2.5 USING CRITERIA

4.2.5.1 What counts as a Criterion?

S4 observed that Criteria differed in the extent to which they embodied value judgements. Some, like the hardware Criterion *low CPU load* were difficult to dispute, whilst others, notably cognitive user-interface related, were perceived as more value laden, e.g. *different button types for different card types helps users*, *exploit visual recognition* and *examples stimulate beginners’ ideas*. S4 was correct in his observation that all Assessments were essentially claims. (It will be recalled that within DRL, all relationships between nodes are contestable Claims). DR notations serve as a vehicle for making the claims within a design explicit; once public, they can then be debated, either by the designers themselves as they reflect on their work, or at a later date by outsiders.

Whilst recognising this, one can still ask if there is any basis to S4’s argument that HCI-oriented Criteria seemed more open to question than system-oriented Criteria. There are two factors, not necessarily mutually exclusive, worthy of consideration: the nature of HCI knowledge, and the cryptic nature of Assessments.

(i) *HCI knowledge*, as wielded by the user interface practitioner at present, is largely ‘craft expertise’ (Long and Dowell, 1989). As experience with written user interface guidelines has witnessed (de Souza and Bevan, 1990), ‘usability rules’ are notorious for being either too general to be interpreted by non-experts, or so specific that a new rule is needed for each problem. Compared to the ‘hard’ design knowledge used by those implementing a system, HCI knowledge is at present ‘soft’ (cf. Newell and Card, 1985; Carroll and Campbell, 1986). Decisions in QOCs which are made on the basis of HCI Criteria may therefore be more open to dispute than other Criteria.

(ii) *Assessments are cryptic* in QOC. It is sometimes hard to understand the reason for an Assessment, because the Criterion is so brief that it is not possible to see how it relates to the Option in question. Very general Criteria (e.g. *easy to use*) are more likely to be queried than focussed ones, because of their ambiguity. S5, for instance, made Decisions on the basis that they were *consistent with rest of system* – a relatively vague Criterion which failed to specify *why* it was consistent. Many other examples of the naming process and problems, are reported in Study 3 [§6.2.3].

4.2.5.2 Ambiguity in the meanings of Criteria

S4 suggested that system maintainers coming to a QOC do not want to know the details of the design process, but simply the clearest representation of the end product. (This is effectively a statement of an underlying principle of the design space analysis approach to DR). S4 went on to argue that the maintenance community's requirements are therefore different from those of the design team; whilst informal semantic relations and terminology in the QOC will be understood within the design team, lack of a more formal basis will not help 'verification of correctness' of designs by outsiders who subsequently modify the system.[3] The use of a project dictionary was suggested, in which commonly used concepts (e.g. Criteria) are defined for the project, as a means of minimising later interpretation problems.

One problem which might arise is interdisciplinary differences in the use of terminology. It is only natural that designers consider problems from their own point of view and expertise, leading to some DRs which are system-oriented, and others more user-oriented. As described earlier, S4's experience with structured software approaches led to interesting reuse of existing QOC structure to maintain consistency when he was adding new QOC [§4.2.1].

Two incidents were noted in which it is likely that the subjects' backgrounds (both computer science) contributed to their use of Criteria. The first was in authoring new QOC: S6 used the Criterion *ambiguous nature of selections* to refer not to possible *user* ambiguity, but to ambiguity for how the *system* should interpret user actions. The second was an interpretational error of existing QOC: S4 interpreted the Criterion *limited memory* to mean system, as opposed to human memory, as was intended.

Bellotti et al (1991) describe one way to represent different perspectives on a design issue. They present an example in which the implementor's interests are dealt with under system-oriented Questions, and resolved by system-oriented Criteria; the user's concerns are addressed through task-oriented Questions and Criteria. Hypertext environments

3 Although a DR is not meant to replace more formal design representations, and as such cannot be used to verify software properties in a formal sense, Study 1 still demonstrated the importance of qualitatively well-structured DR for reuse by others.

should help to reduce the overheads of representing different perspectives (Halasz, 1988; Halasz and Schwartz, 1990).

4.2.5.3 Criterion trees as a support-representation for QOC authors

As detailed in Study 1, Criterion trees were an initial attempt to introduce coherent organisation in the Criteria used. To what extent were Criterion trees useful to subjects in organising and representing QOC?

The design task emphasised the importance of consistency with NoteCards as represented in the existing QOC. In this respect, a representation like the Criterion tree was, not surprisingly, an extremely useful resource, and all subjects used it as a source of Criteria for their own QOCs.

From this initial evidence, there would seem to be an important role for a representation like the Criterion tree, or something similar, which provides an index into the QOC by Criteria rather than Questions. Subjects in effect used the Criterion tree for information retrieval – they needed to find out information about other parts of the QOC. As demonstrated in Study 1, there are certain kinds of query which are extremely hard to answer solely with the Question-oriented structures, for instance:

❑ in which situations did hardware constraints overrule other usability Criteria?
❑ for which problems was processing speed the dominant factor?
❑ which Decisions might be affected if screen-space was increased?

In order for a tool to support queries of this sort, it would need a representation of the links between a given Criterion and all instances of its use, or use of bridging Criteria which it subsumed. Whether the designer actually interacted with a graphical tree in order to issue such a search command, or whether it remained unseen, a representation functionally equivalent to a Criterion tree would be necessary. However, given the need to work with these relationships, it *should* be made accessible to designers, as it explicitly represents the goal structure on the basis of which Decisions in the QOC are made.

4.2.6 MISCELLANEOUS OBSERVATIONS

4.2.6.1 Backing up claims embodied in relationships

A design space analysis explains how a design differs from others, but is it important to understand why a *QOC* is the way it is, i.e. a 'rationale for the design rationale'? QOC's usability and credibility will be damaged if the reasoning behind complex or critical Decisions cannot be adequately communicated.

Provision of 'Backing' argumentation (Toulmin, 1958), allows Assessments to be queried, for instance, when a Criterion is ambiguous [§4.2.5.2]. Meta-argumentation may also be needed to back up other relationships. S7 for example felt it necessary to explain why he was linking new QOC about composite nodes to the existing decision to implement a *graphical browser*. He did this by making what he called a 'meta rationale' comment: *composite nodes are a device for simplifying graphical browser* (Figure 4.6).[4]

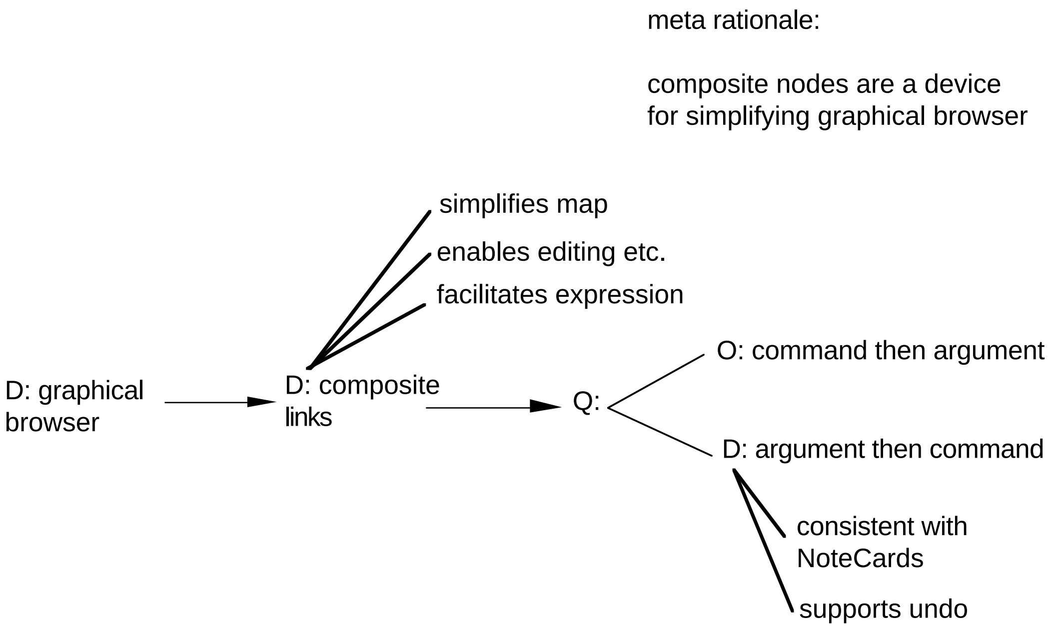


Figure 4.6: Adding 'meta-rationale' to explain why new QOC has been linked to a particular decision (graphical browser).

S7's meta-rationale effectively makes a claim about composite nodes which can now be challenged. If for instance DRL was being used, it would be desirable to encode it explicitly as a Claim. MacLean et al (1991) introduced the *Argument* node for this level of DR. In addition, if S7's QOC had been better structured, a consequent Question would have set the new Option in context. Figure 4.7 shows how these two improvements might be represented.

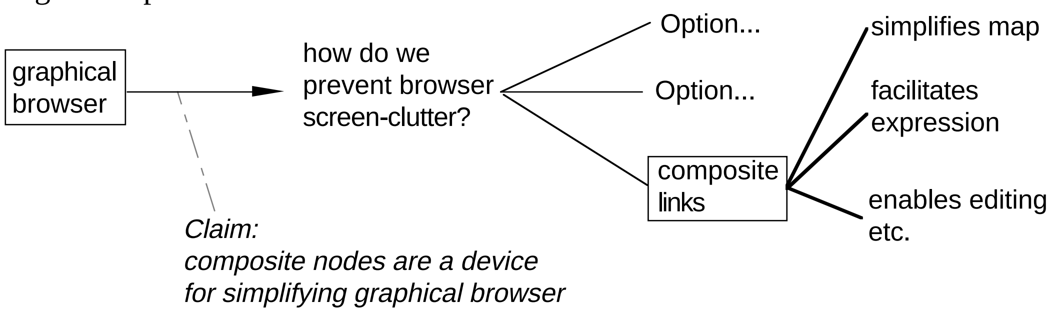


Figure 4.7: Reformulating a consequent Question to set a new Option in context, and using 'meta-rationale' in the form of an Argument to justify adding the Question.

4.2.6.2 Representational form in QOC

4 As Figure shows, S7's QOC was poorly structured, with unbalanced Assessments, and no initial Question setting the use of composite nodes in context. This was one of the poorer structures produced, illustrating the importance of training in QOC.

In a written language, there are strong grammatical rules, and weaker guidelines for appropriate writing styles in different contexts. QOC's grammar constrains the linking of certain node types in certain ways, such that violation of these conventions results in a meaningless representation. However, during the course of the study, it also became clear both from subjects' comments and from analysis of their QOCs, that there were certain guidelines of the second sort – 'groundrules' – about the way in which QOC structures should be represented. As one subject requested, "we need rules for QOC." QOC which conforms to these rules (i) encourages clearer thinking on the part of the designers constructing it, and (ii) is more intelligible to others. In this section, several examples from this study are presented of poorly structured QOC. These point to representational problems which were also observed in Study 3.

S2 used a Yes/No Question as a means of linking in the QOC for his new design, to the existing QOC (Figure 4.8).

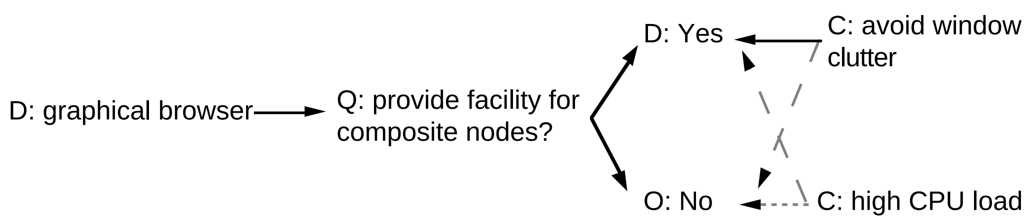


Figure 4.8: Poor QOC structure: use of a Yes/No Question

However, better QOC Questions in this situation make clear what *role* the composite node is playing – what is its *purpose?* S4 and S5 asked themselves this Question, and concluded respectively that it was a *method for reducing screen clutter,* and a new kind of *node.* These are shown in Figure 4.9.

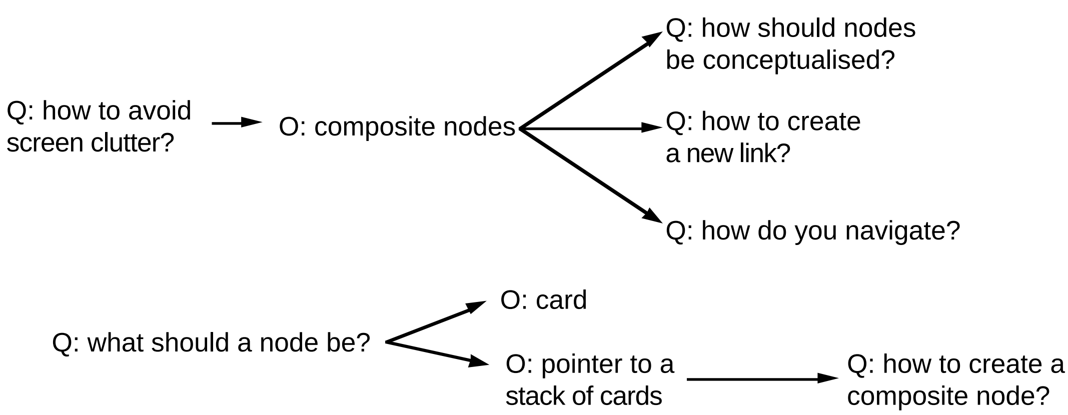


Figure 4.9: Two examples of representing a new idea (composite nodes) as an Option rather than as a Question. This communicates the role which the designer sees composite nodes playing.

Although there is clearly no 'correct' way to conceptualise the role of composite nodes, one might wish to argue further that of the above two QOCs, the second is better structured, since another groundrule is that one should in general not ask direct Questions about Criteria.

One of the benefits claimed for QOC (MacLean et al, 1989) is that use of an explicit representation like QOC exposes the tendency to seek justification for preferred Options and ignore their problems – a QOC graph or matrix clearly reveals Options which have no supporting Criteria. Only one example of confirmation bias (as reflected in the QOC) arose, shown below:

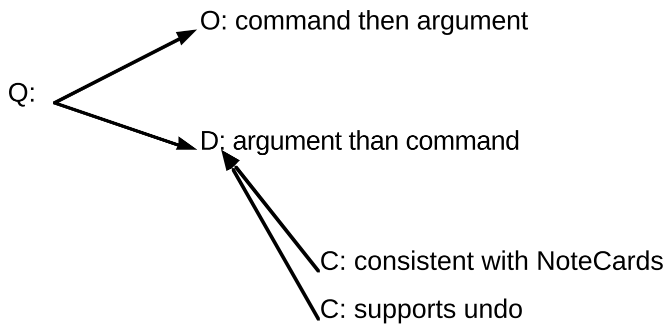


Figure 4.10: Confirmation bias in a QOC structure.

The absence of other examples might suggest that QOC was having the desired effect. Such a conclusion should be treated with caution however, as given the limited time and expertise with QOC, many Options were left unevaluated altogether; how many of them would have received balanced Assessments is unknown.

In addition to the QOCs themselves, a number of comments further highlighted the need for groundrules for representing QOC. For instance, S1 commented that he had come to expect more than one Criterion per Option, and S4, S5 and S6 all expressed the view that a rejected Option should always have an objecting Criterion, so that the reason for its rejection was not left unstated. A heuristic proposed by MacLean et al (1991) covers these comments by proposing that authors should:

❑ *Represent both positive and negative Criteria for each Option.*

S2 and S4 both asked when one should leave Assessments implicit and when to make them explicit; if a Criterion supports one Option, does it object to all the others? This problem only arises when the Question is unfocussed, so that its Options respond to different aspects of the problem. The result is that not all of the Criteria can meaningfully assess all of the Options, and the phenomenon of 'Criterion bunching' occurs, whereby several Criteria assess one Option , whilst the rest assess the other. The principle being violated here can be summarised as:

❑ *A Question should be sufficiently focussed that it addresses only one important issue (so that all Options can provide the functionality, and all Criteria assess all Options)*

Whilst use of QOC should not be regimented so tightly that the flexibility and creativity so critical to design is lost, what may be needed is a ‘styleguide’ which maximises the power of the notation for both author and subsequent reader [see §10.3 for further principles such as these, and discussion].

4.2.7 CONCLUSIONS: MAPPING OUT THE ISSUE SPACE FOR FURTHER RESEARCH INTO QOC AUTHORING

Study 2 was successful in generating data which raised enough issues to merit further study (and in retrospect, it has been possible to cite confirmatory results from Study 3). Having conducted two initial studies, one of QOC retrieval, and the other of QOC authoring, the decision was made to focus on authoring issues. However, the results from Study 1 are still highly relevant, raising a number of requirements for *DR tool developers* to take into account when designing computational support, and for *DR authors* trying to produce intelligible, reusable DR.

Together, the first two studies pointed towards several issues worth pursuing in more detail:

- the overheads of classifying ideas as QOC entities, and integrating them with the rest of the QOC;
- to what extent were authoring problems due to unfamiliarity with QOC and the problem domain?
- QOC’s expressive constraints: does its vocabulary need extending?
- allied to this, what authoring tasks should a QOC software environment support?
- what ‘rules of form’ for QOC are needed – both in overall organisation, and at the level of individual Questions?

Chapter 5

Study 3: Designing with QOC

Part I: *Methodology, and Evaluation of the QOC Tutorial*

5.1 Introduction and experimental methodology

Study 3 was the most extensive of the studies conducted in this thesis. Having identified several key issues relating to the representation of reasoning as QOC, a larger scale study was devised to extend the initial work of Study 2.

The report of this study is divided into two parts. Part I in the remainder of this chapter describes the experimental method followed, the QOC tutorial materials and procedure developed to familiarise designers with DR, and an evaluation of this tutorial. Part II deals with the main results from the study, documenting the key features of QOC authoring which were observed, and discussing the many issues which arose in the course of the analysis.

5.1.1 METHODOLOGICAL ISSUES FOR STUDY 3

In devising Study 3, a pilot study was run to assess the tasks and procedure, reported briefly below. The following experimental design to study QOC authoring was piloted, but rejected for several reasons.

The initial aim was to conduct a large scale between-subjects experiment in order to contrast the effects of using QOC in two stages in the problem solving process, labelled *creativity* (for brainstorming and initial idea generation), and *rationalisation* (for organising as QOC, and decision-making). Three conditions were devised which manipulated the point at which subjects would learn about, and start to use QOC, as summarised in Table 5.1.

		Activities		
Condition		**Part 1: 'Creativity' (30 mins)**		**Part 2: 'Rationalisation' (30 mins)**
'Free creativity'	–––	unconstrained repns no QOC knowledge	intro to QOC	representation of decisions as QOC
'Directed creativity'	intro to QOC	unconstrained repns QOC knowledge	–––	representation of decisions as QOC
'Structured creativity'	intro to QOC	representation of decisions as QOC	–––	representation of decisions as QOC

Table 5.1: Experimental conditions proposed for Study 3 pilot (rejected after initial trials)

It was hypothesised firstly, that the free creativity subjects would be productive in generating ideas, but might have difficulty translating them into QOC, and secondly, that the structured creativity subjects would be impeded by having to use QOC from the start.

The directed creativity subjects were introduced to QOC at the start, but were not constrained to using it until Part 2. They were however encouraged to use it wherever possible, bearing in mind that eventually their decisions had to be recorded as QOC. The goal in this condition was to try to smooth the transition from the unstructured to structured representation of ideas and arguments—to balance the benefits of an unconstrained mode of brainstorming, with the rigour encouraged by the more constrained QOC notation.

Subjects used a Macintosh™ -based drawing package to build QOC graphs, and received no training in QOC notation.

After running three pairs of subjects, it became clear that changes were needed due to difficulties with the user interface, and lack of QOC training; together, these swamped other effects. It also became clear that the study would take far too long to run if reliable quantitative differences were to emerge (at that stage the difficulties in quantitatively analysing QOCs were not appreciated).

Consequently, it was decided that:

(i) more comprehensive training materials needed to be developed to gain more informative data on QOC use, and
(ii) that the experimentally less rigorous, but ecologically more valid observational, video-based analysis should be pursued.

On viewing the data in Study 2, one of the striking aspects was how little design work subjects managed to do. This was due to several factors: limited time (1 hour); lack of familiarity with NoteCards; trying to think aloud or interact with the experimenter; and trying to use QOC for the first time. Coupled with the small number of subjects and variability in background and experience, the amount of design activity recorded was smaller than anticipated, and the occurrence of a given phenomenon was limited to incidents with one, or perhaps two or three subjects.

Within the framework of an observational methodology, certain changes to Study 2's method suggested themselves:

Macintosh is a trademark of Apple Computer Inc.

- study of a design project over an extended period, perhaps over a series of one hour sessions;
- designers should work in pairs or triplets; articulation of ideas becomes natural, and the potential for arguments over different designs increases;
- the task should relate to a system which is more familiar; background knowledge about a system is critical in making the task realistic.
- designers should be much more familiar with representing ideas as QOC.

In Study 3, the task was still limited to one hour for pragmatic reasons (limited time, subject fatigue and availability). However, the other changes were implemented.

An important factor which was pursued to a limited extent in Study 2's conditions, and which from the evidence appeared to be important, was the relationship between the extent of the designer's understanding of the problem, and the properties of the representational notation and medium. This was discussed more fully in reviewing parallels with writing cognition and support [§1.3.2].

Study 3 was thus conducted with the following aims:

- to gather and analyse in depth data on the QOC authoring process (extending the initial work of Study 2);
- to examine more closely the relationship between clarity of design ideas, and the ease with which they can be expressed semi-formally as QOC notation;
- to develop and evaluate QOC training procedures and materials in order to offset very early learning difficulties.

The following sections describe the design task, conditions, training exercises, laboratory and video recording configuration, and procedure for this study.

5.1.2 CHOOSING A DESIGN TASK

5.1.2.1 Pilot study to assess a potential design task

In selecting a task for the final design exercise, certain requirements needed to be met in order to avoid a situation where subjects could agree on the solution immediately, without generating enough discussion material to make QOC useful. The desired property here was ill-structuredness, as characterised by Rittel's 'wicked' problems [§1.3.1.3], namely:

- *complexity of solution*: a major requirement was that the problem should not be so shallow that a design solution could be worked out too quickly; opportunity for simple-minded 'fixits' should be pre-empted.
- *issue transparency*: it should take work to discover and prioritise problems to be resolved; these should also interact with each other sufficiently to force trade-offs.

The first task considered was a graphic design task, to design a public information symbol for an airport which meant “One hour left-luggage office,” without using any written language. In order to assess this task in terms of the requirements set out above, three pilot subjects were video-recorded whilst tackling the problem individually for 30 minutes, thinking aloud.

Results and conclusion

The key problems in this task are representing *left* luggage, that it must be collected at most *after one hour* has passed, and the fact that it is *safe.* From this brief evaluation, the main problem turned out to be that subjects were able to identify the issues and relevant criteria too easily—they were too transparent. Whilst solutions differed considerably between the pilots, the range of possible arguments seemed limited.

It was decided that this task would make suitable training material, and it became Training Exercise 1 (Appendix 5), in the form of a hypothetical design dialogue which subjects represented as QOC. Although not an original requirement, it was decided that a *software* design problem would be highly preferable, given that QOC was developed for representing user interface designs in particular.

5.1.2.2 The ATM design task

The problem selected for this study was the design of a bank’s automated teller machine (ATM). ATMs are now a familiar piece of technology to the general public, and as such avoided any domain familiarity problems for subjects. However, whilst their purpose and general mode of operation is understood, their potential functionality, and the range of possible user interfaces is sufficiently complex that designing them is a non-trivial exercise. MacLean et al (1990, 1991) also report studies using the ATM problem.

5.1.3 CONDITIONS

Analysis following Study 2 led to the hypothesis that the more developed design arguments are, the easier it is to translate them into QOC notation. That is, the clearer one’s ideas are, the more likely it is that (i) they can be summarised (they can be given *names*), their status and functional roles are known (they can be *classified*), and their relationships are clearer (they can be *structured*). In this study, although the decision had been taken to follow an observational rather than a parametric experimental design, two conditions were run in case effects of the ‘state of development of ideas’ could be detected, even without exercising strict controls.

The variable was manipulated by varying the amount of detail presented in the ATM design problem statement. In the ‘upstream’ condition, the problem statement was brief, and no detailed cues were provided as to the relevant issues or criteria to consider, thus simulating the situation of designers just beginning to generate ideas over a new problem. In contrast, the ‘downstream’ problem statement proposed an alternative design for consideration, and provided additional information on problems with the existing design, cueing subjects to pertinent issues, design alternatives and trade-offs. This condition represents a situation further downstream in analysis of the problem, where there is already some understanding of the problem or class of problem – the key issues are clearer, and a body of ideas now needs to be organised. The two problem statements are reproduced in Appendix 8.

5.1.4 SUBJECTS

Subjects were recruited (i) through advertisements within the university campus, (ii) through a collaborative agreement with the information technology division of a local company, and (iii) through personal contacts with designers. In total, 24 experimental subjects were studied, 12 per condition, with two pilot subjects, whose data were not used in the analysis. The experimental subjects comprised 7 undergraduates, 14 software professionals (2-12 years experience), and 2 computer science research staff. A regular computer user who expressed an interest in user interface design issues was also used (Pair 7).

Students and professionals were balanced between conditions as far as possible. With the exception of the designers from the local company, all subjects were paid for participating. Subjects were trained, and designed in pairs. With the exception of Pairs 6 and 7, each subject knew his/her partner in a work or social context.[1] Subjects’ backgrounds, design experience, and assignment to condition are shown in Table 5.2.

1 Pair 6 worked efficiently together, with no problems arising from being strangers. However, the non-designer member of Pair 7 created an imbalance, which resulted in much of the work being done by his partner.

Pair	Occupation & design experience	Occupation & design experience
1	computer science student (3yrs)	economics student (3yrs)
*2	computer science student (3yrs)	computer science student (1yr)
3	computer science student (3yrs)	computer science student (1yr)
*4	computer science student (1yr)	programmer (1.5yrs)
5	programmer/documentation (2yrs)	freelance software engineer (15yrs)
*6	systems support programmer (5yrs)	Info. Processing M.Sc. student/ some commercial programming(1yr)
7	office applications user (10yrs)	freelance programmer (10yrs)
*8	analyst programmer	analyst programmer
9	operations research project manager (16yrs)	operations research scientist (3 yrs)
10	systems analyst (12 yrs)	office and business systems analyst (3.5yrs)
*11	senior analyst programmer/system analyst (10yrs)	analyst programmer/systems analyst (4.5 yrs)
*12	computer science research associate (4 years research in software engineering & interactive systems design)	computer science/HCI lecturer (5 years research in HCI)

Table 5.2: Subjects in Study 3.
*= Pairs who received the Downstream problem statement. (Pair 10 were accidentally assigned to the Upstream condition, so Pair 11 were assigned to Downstream).

5.1.4.1 Note: The notation used in this study

As a consequence of the difficulties observed in Study 2, the decision was taken to try to further reduce the representational overheads by using Issues instead of Questions. It was hypothesised that an Issue could be more quickly jotted down compared to formulating a question.

5.1.5 TRAINING EXERCISES

QOC is relatively simple to grasp in terms of the basic roles which each node and link type plays; it is in actually *using* the notation to represent ideas which Study 2 indicated caused problems. The training schedule introduced designers initially to representing other people's design ideas in a number of different contexts, and then their own in the ATM task. Each exercise is briefly described below, summarised at the end in Table 5.4.

5.1.5.1 Training exercise 1: QOC analysis of mini design-discussions

After the introduction and some initial examples, subjects (working on their own) tackled the task of representing the Issues, Options and Criteria present in fictional extracts of design discussion. They completed, in 20 minutes, as many examples as they could from a set of ten (Appendix 4), after which the experimenter checked and discussed their 'answers' against his own. One of the worked examples is shown in Figure 5.1.

> - pop-up menus are probably the best thing to go for - click anywhere on the screen with the right-hand button, and it appears instantly at that point. No need to move the mouse up to the top of the screen everytime.
> - the problem is with the hardware folk - they've said they want to try a cheaper one-button mouse – we may be forced to go for pull-down menus at the top of the screen.

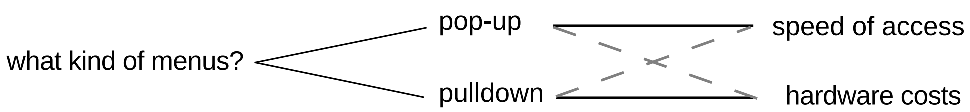


Figure 5.1: A worked example from the first QOC training exercise.

This exercise introduced subjects to a number of 'QOC phenomena' which they might meet in the future, that is, common representational tasks or problems in translating discussion into structured QOC. These included:

- decomposing an Issue into two subIssues
- finding the right level of generality, particularly with Criteria
- framing Issues in a way which accurately captures the problem
- recognising that a Criterion which supports all Options can be embedded in the Issue statement, because they implicitly respond to it as a requirement

Thus, on completing this exercise (about 40 minutes), subjects had been introduced to and discussed the fundamentals of representing DR as Issues, Options and Criteria. Subjects were not video-recorded during this exercise.

5.1.5.2 Training exercise 2: QOC analysis of extended design dialogue

This exercise presented subjects with an extended version of the mini-discussions in Exercise 1. A dialogue was made up between two designers discussing the graphic design for a public information sign. The 'transcript' of this dialogue was given to subjects, together with sketches of the sign which accompanied their ideas (see Appendix 5). The task was to represent the content of the discussion in QOC. Subjects worked in pairs on this for 30 minutes, and were video-recorded to familiarise them with being on camera, and to collect initial data on very early use of QOC.

5.1.5.3 Training exercise 3: QOC analysis of scripted-video dialogue

The final training exercise before the ATM task made use of a procedure devised as a means of exposing subjects to excerpts of realistic design discussion, whilst allowing the experimenter to maintain complete control over the content of the discussion. Subjects were shown video extracts of two actors following a design script (which comprised both dialogue and sketches for the design of a video-recorder remote control – Appendices 6 and 7). The task was to produce DR representing the decisions made and the underlying reasoning. Subjects were thus placed in the secretarial role of 'QOC scribe,' as one might imagine at a design meeting; they were required to work harder than in the previous exercises, in that they had to extract possible Issues, Options and Criteria from the discussion in real time.

The scripted video enabled four issues to be studied: (i) subjects' ability to capture the main content of a real time discussion as QOC; (ii) the evolution of QOCs over time, as the design evolves; (iii) the process of merging two QOCs into one; (iv) the effect of the way in which ideas are expressed in a discussion, on its subsequent representation as QOC. Details of these are described below.

Subjects were allowed to take rough notes of any sort during the video, which was in two parts. Timing details are shown in Table 5.3, and explained below.

Duration hr/min	Task
0.05	introductory instructions
0.07	working separately, watch Video Part 1, and take notes/create QOC
0.10	represent notes as QOC
0.14	watch Video Parts 1+2 without a break, and take notes/create QOC
0.10	represent notes as QOC, revising Part 1 QOC as necessary (using different colour pens from Part 1 QOC to show changes)
0.30	work together to merge QOCs into a single best QOC of the design session – sketches from video provided for reference (video recorded)
Total 1.16	

Table 5.3: Training procedure in scripted-video exercise

Subjects had three opportunities to revise their QOCs: (i) during the periods after each video clip, when they could edit any QOC created whilst watching the video, (ii) whilst editing their Part 1 QOCs in the light of Part 2, and (iii) in the process of collaboratively merging their individual QOCs in the last stage. It was hoped that differences might emerge in the use of QOC, leading to the restructuring in the light of either contrasting interpretations of the argumentation, or views of how to represent it – for example, different ideas about what the key Issues or Criteria were.

The last factor of interest was the possible effect which the actual way in which ideas are expressed has on the 'QOC notetaker.' The script was constructed such that there was a range of ways in which Issues, Options and Criteria were expressed. Thus, an Issue could be raised as a straight question, the most common way in which they are represented in QOC (e.g. *how do you switch to program mode?*) – this was considered to be the easiest to translate as QOC. However, Issues could also be implicit, when combined with an Option, or an Option and Criterion (e.g. *we've got to have a remote control, as bending down's out of the question*) – to represent this as QOC requires generation of a suitable Issue (e.g. *what input device?*), and separation of the Option from the Criterion. Similarly, an Option can be expressed as though it were an Issue (e.g. *why not use largish keys, with different colours; is it possible to reuse the tape control keys?*), and a Criterion as an Issue (*will they be able to read it?* [legibility]*; how can you move around the table?* [ease of navigation]).

Attempting an analysis of this sort carries with it the assumption that there is such a thing as a 'correct' QOC interpretation for a given idea – that, for instance, certain ideas clearly should be Options. This assumption rested on the belief that there are principles for well structured QOC which communicates the issues in a clear manner, and which supports good design problem solving. Candidate properties of such QOCs were collated over the course of the research, and are summarised in the general discussion [§10.3].

In summary, the dialogue was constructed with these 'traps' because real discussion in all its richness clearly is not limited to assertions in the vocabulary of DR notations.[2] To this end, the hypothesis was proposed that superficial expressive form exerts an influence over the QOC representation (e.g. a question about something will be directly translated into QOC as a Question).

5.1.5.4 The design exercise: ATM design using QOC

The rationale for using an ATM design task was presented earlier, and the design problem statement for each condition is reproduced in Appendix 8. Subjects were given five minutes to read through the statement on their own, making any notes they wished, and querying anything that was not clear. They then had one hour together (video-recorded) to discuss the ATM's design, summarise their new design in terms of what the user would do on stepping up to the machine, and represent the design rationale behind the decisions they had made. They were allowed to use rough notes, sketches or any other representation they felt useful—it was emphasised by the experimenter that they were not confined to recording only QOC structures.

---

2 This is the case at least in normal discussion between designers who know nothing about DR. One might hypothesise that designers trained in a DR notation will adapt their language for whoever is recording it, to make clearer the functional roles of their contributions to an argument. (This mode of working is central to Conklin and Yakemovic's proposed use of gIBIS [§2.1.1]).

5.1.6 MATERIALS AND RECORDING CONFIGURATION

5.1.6.1 Representational media

The pilot study [§5.1.2.1] showed that use of drawing tools to represent QOC created learning problems for subjects which outweighed any benefits over pen and paper. As a result, all notes, sketches, DRs, and other representations created by subjects were on A3 sheets of paper. The advantages of using paper-based representations were noted in Study 1 [§3.1.3.3.5].

5.1.6.2 Video recording configuration

The recording configuration used in this study is shown in Figure 5.2. The lab was divided in half, and subjects changed position depending on the exercise they were engaged in. When working separately, one subject sat in each room, and in the case of the scripted-video exercise, studied his/her monitor. When working together, subjects used the camera room and the experimenter observed them via the monitor in the other room. When subjects were actually being recorded together, their monitor was not switched on (i.e. they could not see themselves). This was partly for technical reasons (monitor interfered with microphone), but also because it was felt that it would be off-putting to the subjects. The video image was thus an 'over the shoulder' view of the table between the two subjects.

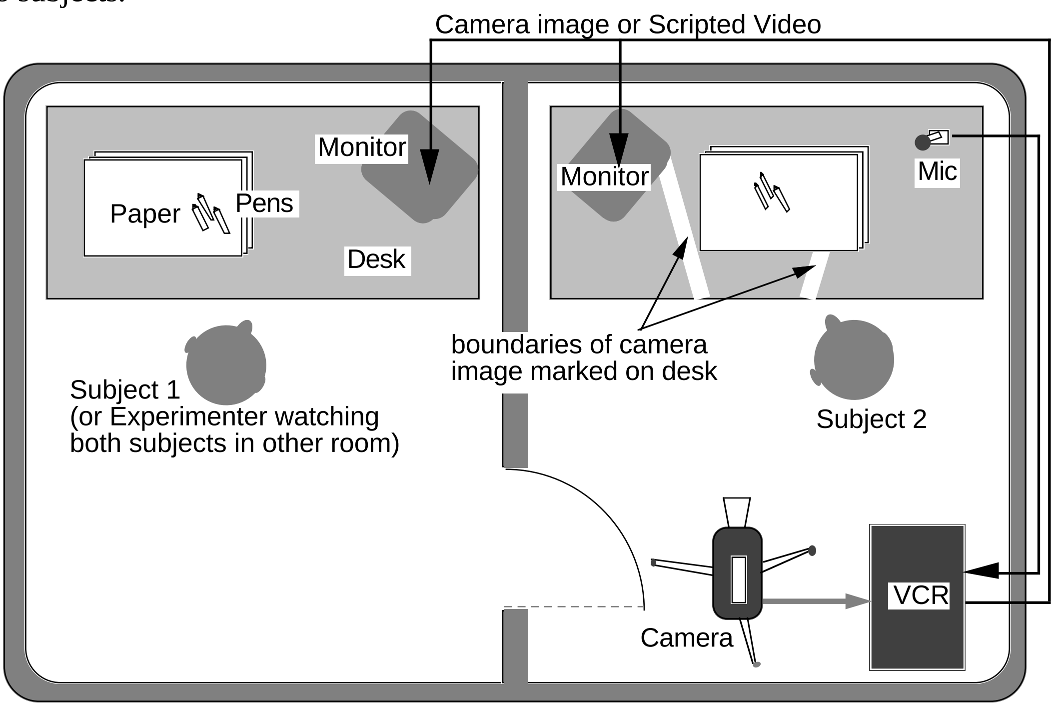


Figure 5.2: The video-recording configuration used in Study 3.

5.1.6.3 Intrusion on subjects by video recording

Ideally, images from two cameras would be mixed, one to capture gestures and activity away from the main focus of attention (e.g. one subject making notes while the other creates QOC), and the other to record the main representational activity close-up (e.g. Tang, 1989). However, due to limited hardware resources, recording was limited to a close-up shot as above, and subjects were asked to make a few minor allowances in the way in which they worked to facilitate the recording. One was to keep representations within the boundary lines marked on the desk (Figure 5.2). This included not only the representations they were currently working on but any to which they referred verbally (e.g. their own QOCs when producing the combined QOC in the scripted-video exercise).

Subjects were also asked to think aloud as far as possible, to minimise the occurrence of long periods of silent inactivity. However, the advantage of working in pairs is that there were relatively few ‘long silences.’ As a result, articulation of ideas did not seem forced, and the above impositions on subjects did not intrude to such an extent that the use of video-based observation became obtrusive – one of the main advantages of the technique.

5.1.7 Summary of training procedure

The procedure for QOC training and subsequently tackling the ATM task took about 4 hours in total. Recorded use of QOC amounted to 24 hours in total, plus debriefings. In some cases subjects completed the scripted-video and ATM exercises on a subsequent day, but the majority were able to finish within a morning or afternoon, with a break before the scripted-video. Pragmatically speaking, therefore, the training schedule is comparable in duration and intensity to many of the commercially run half-day workshops which computing professionals attend.[3] The overall procedure is summarised in Table 5.4.

---

3 Indeed, a shortened version of this training schedule was presented to a software project team at a local company interested in design rationale (Case Study 3).

Exercise	Mode of working	Duration	Videoed
❑ Introduction to DR concept and QOC notation – two short examples	subjects work with experimenter	20mins	no
❑ 10 practice examples; check answers	subjects work separately [20mins], and then check QOC with experimenter	40mins	no
❑ QOC analysis of a written design dialogue; check answers	subjects work together [30mins], and then check QOC with experimenter	45mins	yes
Break			
❑ QOC analysis of a scripted design session on video	subjects separately watch video and create QOC [45mins], then collaborate to produce single QOC [30mins]	1hr 15mins	yes (final session only)
❑ ATM design task	subjects work together	1hr	yes

Table 5.4: Overview of procedure for training and evaluating designers in the use of QOC.

5.2 Approaches taken to analysing the data

This section focusses on how the large amounts of observational data generated in this study were analysed, both by the more usual qualitative techniques, but also quantitatively. The emphasis therefore is on the *process* of abstracting useful information from the data; the significance of this information is dealt with subsequently in the Results and Discussion (Chapter 6).

5.2.1 QUALITATIVE ANALYSIS OF DESIGN ACTIVITY AND DESIGN RATIONALES

A lot of raw data was generated in the form of video recordings, design rationales, sketches, and notes. The steps by which this analysis was carried out are described below as they may be of interest to other researchers planning studies of a similar nature.

5.2.1.1 Transformations of data in qualitative analysis

From the initial recordings of subjects' dialogue and design activity, the data underwent several transformations. If we take the speech and concrete representational activities of subjects as the sources, Figure 5.3 summarises the process involved. In each transformation, certain kinds of information are filtered out (shown by a ⇒) to yield a different representation, which itself then makes explicit (+) new information.

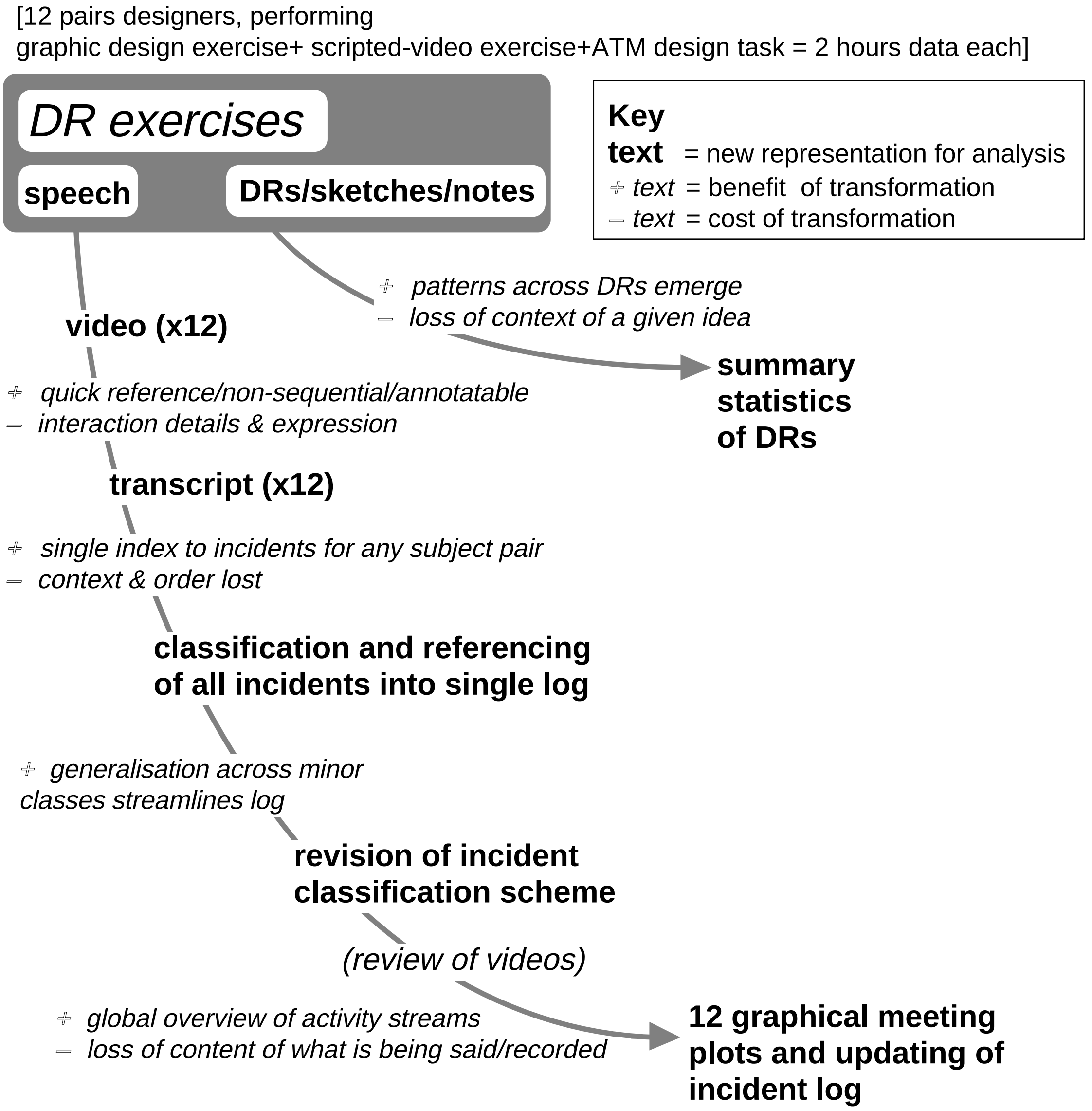


Figure 5.3: Steps in qualititative analysis of data in Study 3.
Taking designers' discussions and representational activities as the source, representations of each underwent a series of transformations revealing new aspects to the sessions, whilst hiding others.

5.2.1.2 Graphical meeting plots

The final transformation shown above is the generation of graphical meeting plots. The other representations fail to offer a global view of the patterns of activity in a session, making it difficult at times to locate incidents, or grasp how the different streams of activity relate to each other. A section of meeting plot (adapted from Olson and Olson 1991) is shown in Figure 5.4, illustrating how discussion, sketching, notetaking, design rationale, and other activities relate over time.

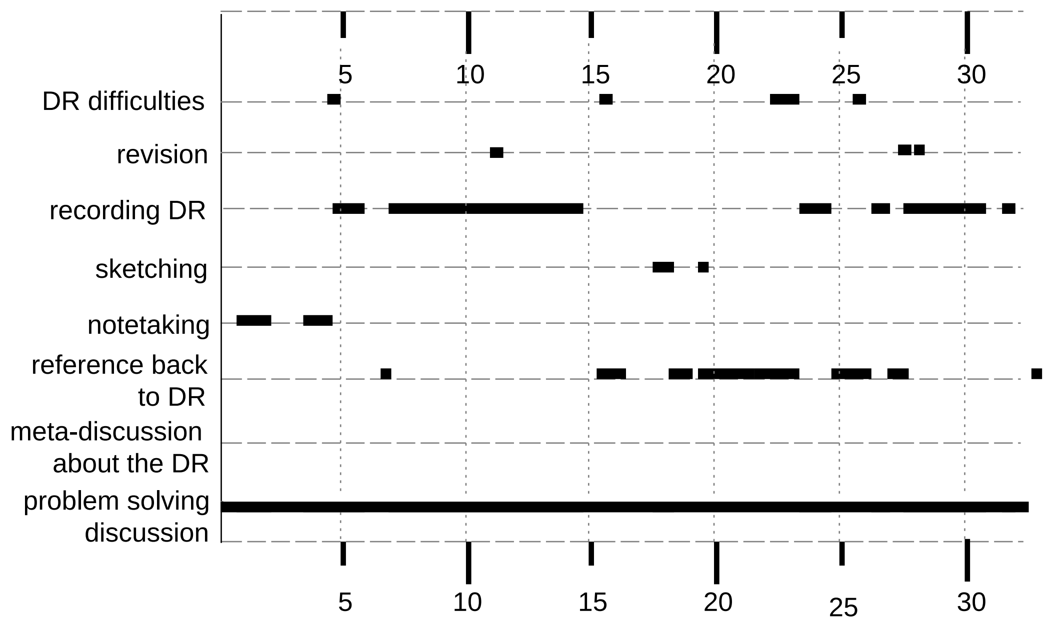


Figure 5.4: Part of a graphical meeting plot, which shows which activities designers are engaged in at any moment.

A meeting plot is generated by defining axes for each activity of interest, and tracking each activity whilst watching the video. It is common to find that as the first plot is constructed, new activity streams suggest themselves, whilst others become redundant. Setting up meeting plots is therefore an iterative process. This is also the case in logging classes of incident – as Figure 5.3 shows, the incident log was revised as classes were refined and deleted.

It will be noted from Figure 5.4 that *problem solving discussion* seems to proceed continuously, as signalled by the continuous black line; whilst showing that task-related discussion was persistent, it offers no clues to the content of that discussion. The point is that the categories tracked through a meeting can be made as specific or general as necessary. Analysis of the same session by another researcher who was interested in, for instance, conversation analysis rather than QOC authoring, might refine the granularity of *problem solving discussion* into many subclasses, collapse the streams for *QOC, sketching,* and *notetaking* into a single one labelled *recording,* and introduce new activity streams for gesture. The meeting plots proved useful as an index into the video data.

5.2.2 QUANTITATIVE ANALYSIS OF DESIGN RATIONALES

The design rationales produced by subjects are amenable to quantitative analysis for summary purposes. Numbers of Issues, Options and Criteria can be logged, relative proportions of one to another calculated, and a number of other interesting measures derived, such as the number of decisions made without considering negative Criteria, or the number of reused Criteria.

The informality of QOC permits designers to interpret the content of discussions in more than one way, as documented in Studies 1 and 2. A consequence of this is that quantitative measures of 'correctness' of QOC are difficult to derive, and in all cases require a degree of licence on the part of the analyst. For instance, an attempt was made to evaluate the QOCs in the scripted-video exercise against a canonical QOC. In doing so, numerous difficulties were encountered in scoring: Options and Criteria were expressed at different generalities from the 'answer'; Options appeared under different Issues; Issues appeared as Criteria; subjects added new Criteria, or evaluated decisions differently from the script in the video; nodes were phrased differently – were they equivalent?

In sum, analysis of *content* in QOCs is fraught with difficulty; for this reason, the quantitative analyses have been limited primarily to *syntactic* regularities, that is, logging phenomena which appear at the structural level of QOC notation (reported in §6.1.2).

Let us now turn to the results. In the remainder of this chapter, aspects of interest from the training exercises are briefly described, focussing on the influence of variables. Chapter 6 reports the main characteristics of QOC authoring which emerged, illustrated with examples from both the training exercises and the ATM design session.

5.3 Evaluation and discussion of the training exercises

From studying the design rationales and associated behaviour of subjects during this study, it became apparent that the training exercises possessed a number of interesting characteristics, in terms of the QOCs produced. This section reports the results from the training exercises, assessing their success, and commenting on issues relevant to the development of DR training materials in the future.

5.3.1 Training exercise 1: QOC analysis of mini design-discussions

This was the first opportunity subjects had to use QOC, and as such the data represent very early stages of QOC use. Overall, subjects learned the basic notational conventions quickly, and were able to give fair QOC translations of the brief extracts. However, a number of errors and difficulties did occur, including:

- failure to assess all Options with each Criterion;
- expressing Criteria negatively instead of positively;
- formulating Yes/No Issues about single Options instead of Issues which permit a range of Options to be considered;
- using unfocussed Issues which raise more than one important point.

These kinds of representational problems are now recognised as violations of certain rules for 'well-structured' QOC, which were documented in the course of the studies, summarised later [§10.3]. These representational problems remained with designers for the remainder of the study, suggesting that only a small improvement, if any, occurred in their ability to produce 'polished' QOC. Indeed, the *process* of refining Questions, recognising unevaluated Options, and so forth, is a critical part of developing an understanding of the design space. Note however that these observations relate only to the use of QOC to represent other peoples' discussion; subjects had yet to learn to represent their own ideas during design.

5.3.2 Training exercise 2: QOC analysis of extended design dialogue

In this exercise subjects were placed in a position analogous to a designer contemplating the results of a meeting, with a view to summarising the key issues which emerged (in the design of the public-information symbol); the DRs were as a result more retrospective than narrative [§2.1.1, §2.1.5], that is, they were constructed with all the arguments to be represented available in advance.

After the exercise, the experimenter went through a possible QOC analysis which had been prepared beforehand. This QOC was intended to be an example of how to represent the design space systematically. It was found that most of the subjects' grouped the Options under similar Issues to those in the prepared QOC. This overall consistency reflects the dialogue's relatively well-delineated Issues. Nonetheless, there were some interesting variations between subject pairs, due to differences in Issues and in the use of Criteria. These are discussed below.

Issues varied along two dimensions – the view they offered of the design space, and their granularity. Granularity refers to the extent to which subissues are explicitly decomposed and represented as Issues, or left implicit in a more general Issue. When representing one's own ideas, the grain of analysis and representation often reflects the subjective importance of the problem, and one's expertise in the domain. In this exercise where the ideas are not subjects' own, variation in granularity reflects individual differences in the importance each pair attached to an issue – should it be made explicit or not?

For example, the solution QOC asked a general question, *How to show security of luggage?*, with Options *show case in safe*, and *show an official*; the following Issues then asked how to implement each of these Options (top half of Figure 5.5). However, Pairs 2, 10, and 11 grouped these together into one Issue covering all the Options. Pair 2's Issue and Options are shown in the lower half:

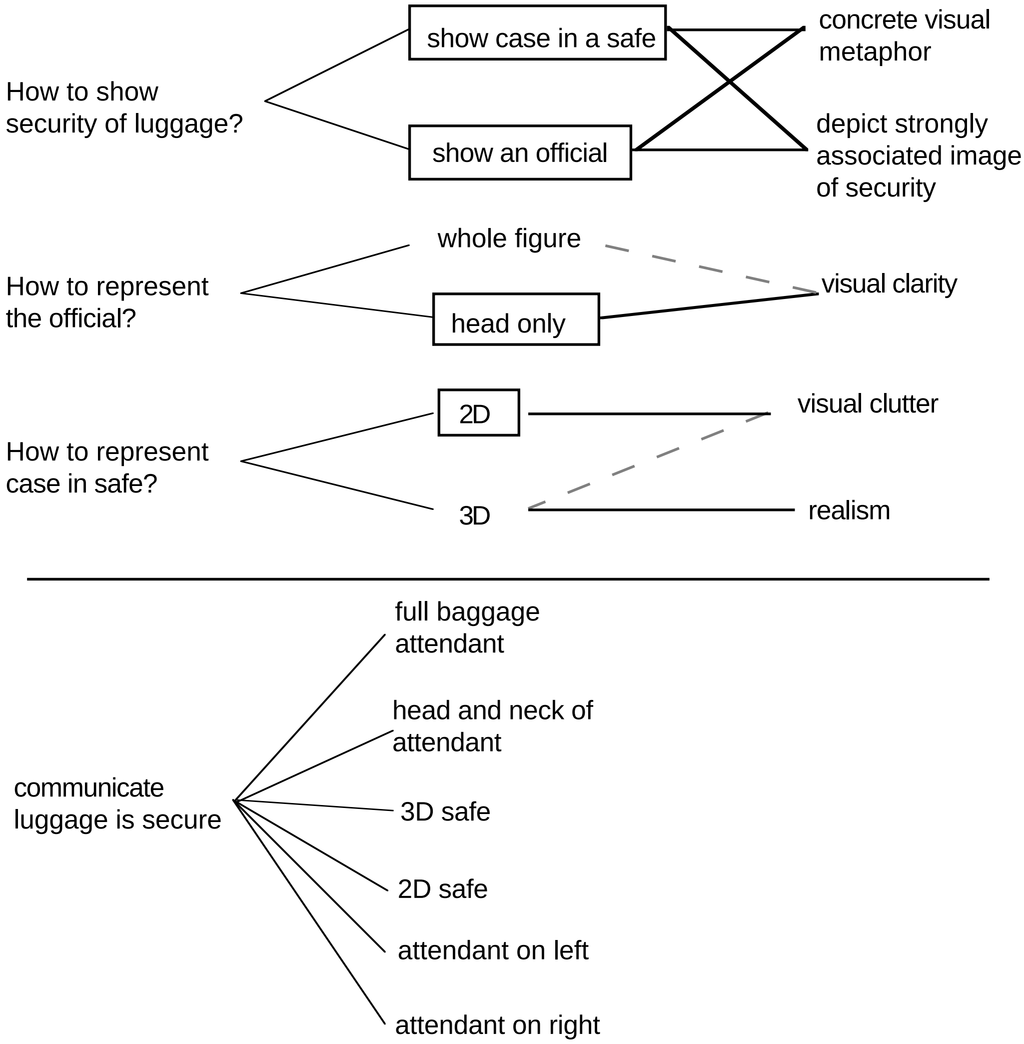


Figure 5.5: Granularity differences in Issues. Options can be grouped under a more general Issue, or under more focussed subIssues.

Although Pair 2 did not have time to record Criteria, it can be seen that if they had done, the resulting QOC would have been confusing. The three subissues are embedded in their single Issue, each with a pair of Options. Decisions between these Options would be made on the basis of *relative* assessments *within* each pair (e.g. *3D safe* is *more realistic* than *2D safe*); comparisons *between* pairs is inappropriate (e.g. *realistic* is not relevant to making decisions about the other two subIssues). A more general Criterion like *clarity* could conceivably be used to assess all of the Options, but its meaning would vary for each pair (e.g. *clarity* meant *visual distinctiveness* in relation to the design of the attendant, but *clarity of meaning* in relation to his position).

To summarise, Issues which are too general spawn Options which represent slightly different parts of the design space; as a result, it becomes difficult to represent their evaluation, because Criteria are either relevant to only a subset of the Options, or take on different meanings when applied to different subsets of Options.

Issues differed not only in granularity, but in the orientations they took in grouping Options. *Granularity* can be viewed as essentially the same view of the design space, but varying in the level of detail. Differences in the *orientation* of Issues implies qualitatively different ways of thinking about Options—alternate ways of structuring the space. Whenever a new Option arises, designers must ask themselves what Issue it relates to. Difficulty arises however when there is more than one Issue under which it could be placed.

Consider some concrete examples. Under which Issue should the Option *show suitcase in 3D safe* be placed? Pair 3 focussed on the fact that it involved *baggage*, and under a general issue of *baggage design,* had Options *many bags, single bag, bag in safe (3D), bag in safe (2D)* . Pair 9 followed a similar line, but used two Issues, making the number of dimensions (2D/3D) into a separate Issue (Figure 5.6):

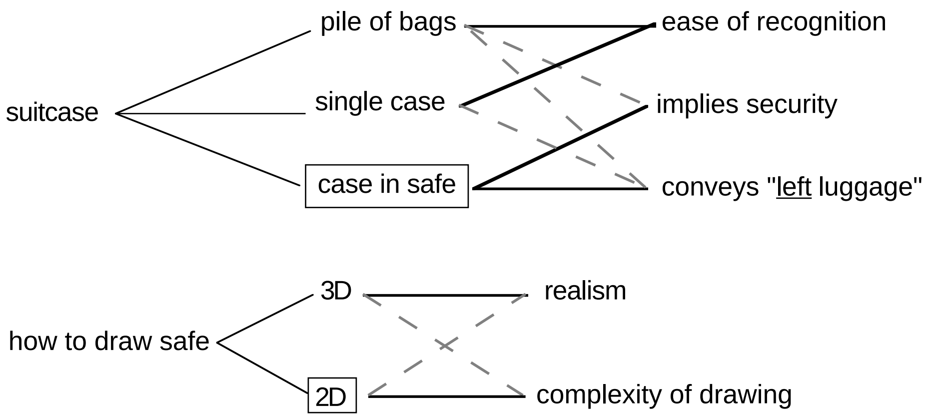


Figure 5.6: Using an Issue to group all Options with a common feature (representation of suitcase)

However other pairs decided that *security* was the important property and placed *case in safe* under an Issue about conveying security. A similar pattern arose with the Option *show people in a queue.* It is again possible to decide that *people* is the important feature to use as a grouping principle, or that the *function* of showing a queue should be highlighted (to indicate waiting for something), which lead to an Issue asking how to show that the luggage was left somewhere.

To conclude, the extended dialogue provided subjects with their first experience of creating a retrospective QOC. In the process, the main representational decisions they had to make were Issue granularity, and how to interpret the function of certain Options. The variety of QOC produced, some of which was quite poor, demonstrated the need for more training and practice in using QOC. If the QOC is ambiguous, (i) it cannot be reused by outsiders, and (ii) the potential benefit to the designers is significantly weakened, because thinking is unfocussed.

5.3.3 Training exercise 3: QOC analysis of scripted-video dialogue

Analysing the dialogue in the scripted-video was the third, and most complex training exercise subjects performed. This procedure explored how well real time discussion could be captured and represented as QOC, whilst exercising a degree of control over the development of the QOCs over time, and the process of merging QOCs. These different interests are discussed in turn.

5.3.3.1 Changes to the QOC over time

The development of QOC was monitored at two points: changes to QOC after watching Part 1 of the scripted-video for the second time, and the process involved in merging the two individually authored QOCs. Figure 5.7 shows the 'editing profile' for QOC between the two viewings.

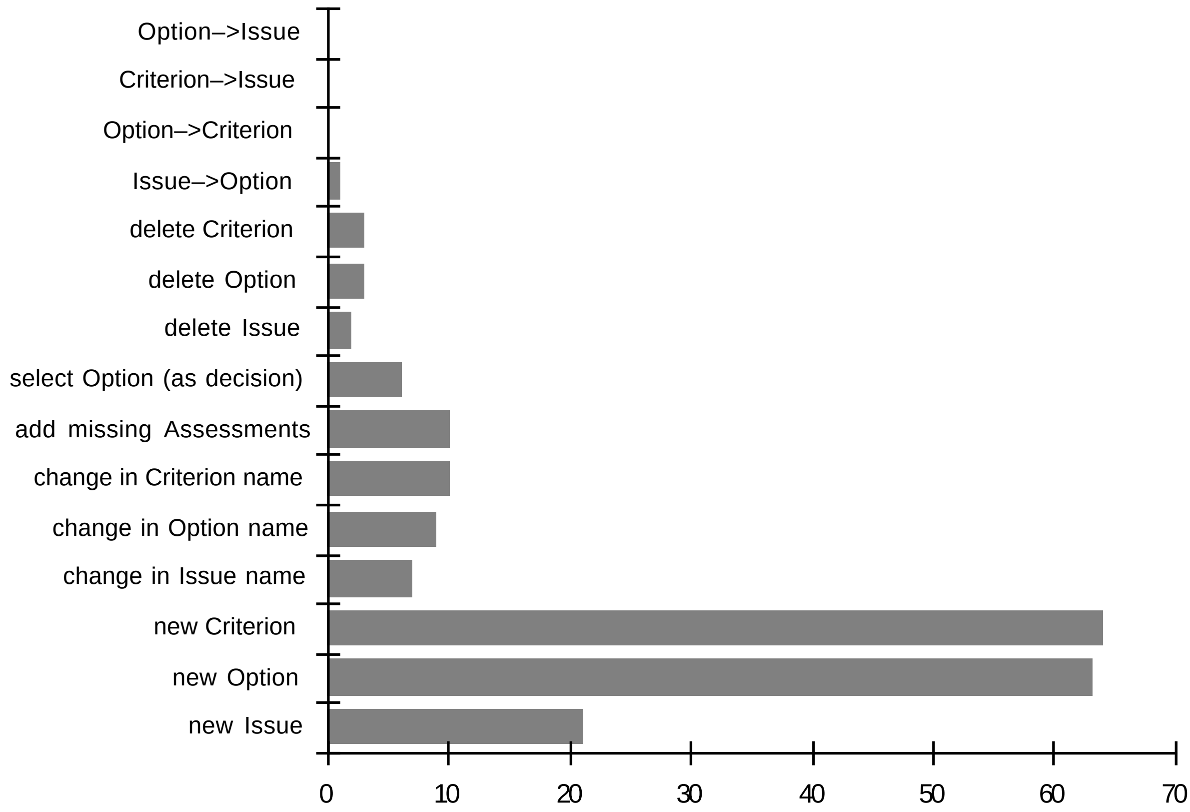


Figure 5.7: Changes made to QOCs over two viewings of the first half of the scripted-video(totals for each measure).

The data show that a substantial number of Issues, Options and Criteria were missed in the first pass QOC, accounting for 74% of the revisions. Modifying the names of nodes accounted for half (51%) of the other changes. Other activities, though less frequent, included filling in Assessments, and marking decisions which had been left open up to that point. Due to unforeseen circumstances, Pair 12 did not have enough time to undergo the full training procedure, and underwent a briefer version. As a result they did not perform the graphic design exercise, and viewed Part 1 of the video only once. Discounting this pair, only two other subjects (separate pairs) left their Part 1 QOC untouched.

The number of new nodes alone indicates that it was impossible to record all of the important information as QOC in the 7 minute Part 1, at least within the constraints of the procedure used here (allowing 10 minutes to translate rough notes into QOC). With regard to the two other main activities shown in Figure 5.7, (i) the prevalence of *renaming* supports the evidence gathered from other tasks as well [§6.2.3] showing that naming is an important, dynamic activity, and (ii) the finding that Assessment links were omitted during the first-pass QOC is consistent with the fact that expressing the strengths of trade-offs can be difficult [§6.3.4].

As Figure 5.7 shows, there was very little restructuring activity in this exercise. It is most likely that the limited time available precluded restructuring the QOC, since this often comes out of reconceptualising the design space, which in turn requires time and effort. Case Study 2 drew similar conclusions [§8.2.2.2.2, §8.2.6].

5.3.3.2 Effect of verbal expression on QOC representation

One hypothesis tested was that the way in which ideas are expressed during discussion can affect the manner in which they are recorded as QOC. Use of the scripted-video enabled the experimenter to manipulate the way in which ideas were expressed.

Analysis of QOCs in this exercise failed to reveal many examples of verbal expressive form affecting QOC form. It would appear that generally, subjects did not directly transcribe from the surface form of the spoken idea to the equivalent in QOC, e.g. questions becoming Issues, or ideas expressed as ‘alternatives’ becoming Options—initial expressive form seemed to be ‘filtered out’ by the time rough notes had been transcribed into QOC. However, four subjects made Issues out of Criteria expressed as questions, and one made an Issue out of an Option expressed as a question – that is, five incidents support the hypothesis that the way in which ideas were expressed affected their subsequent representation as QOC.

Overwhelmingly, QOCs for this exercise were very similar, formulating and answering similar Issues in similar ways; in one sense, it would be surprising if this were not the case, as it would indicate widely differing interpretations of the video. The small effects of verbal expression suggest that QOC is relatively impervious to such surface level features as form of expression, and instead, the deeper meaning of ideas and their relationships predominate; as the ideas are marshalled into a coherent structure, initial form of expression is lost. This appears to be the case at least when the ideas are reflected upon;

in the case of more narrative DR (such as gIBIS), when ideas are encoded 'on the fly' during meetings with little of no time for reflection or editing, it may be that form of expression has a greater effect on the representation – however, there are no analyses available of sufficient detail against which to compare these results. Differences which did arise between subjects are more attributable to granularity than to different views of what the Issues were, and to individual preference for making Issues out of particular Criteria, rather than to different ideas as to what the Criteria were.

5.3.3.3 Analysing the merged QOC

In order to assess the changes, if any, which the independently authored QOCs underwent as they were merged, the merged QOC was analysed with the following questions in mind:

- *sources*: From which of the source QOCs was each Issue and associated Options and Criteria drawn?
- *transformations*: Were the source QOCs transformed in any way during merging?
- *size*: What was the relative size of the merged QOC to its source QOCs?

Three *sources* of Issues in the merged QOC were identified:

- *combined source*: The Issue and associated Options and Criteria in the merged QOC were a combination of elements from each of the source QOCs.
- *single source*: The Issue and associated Options and Criteria were taken largely or completely from one source QOC; the other source QOC either had no representation of that Issue, or used a representation which contributed nothing to the merged Issue.
- *dual source*: The Issue and associated Options and Criteria were identical in each source QOC, and were imported into the merged QOC without modification.

With respect to *transformations*, the merged QOC was checked to see if Issues, Options or Criteria had been *improved* by being made more *focussed* or more *general* relative to the source QOCs.[4] The transformational measures thus represent improvements to the QOC. Instances of poor transformations were also logged (under *poor/poorer QOC*).

---

[4] These classes were defined with respect to *improvements*, because level of abstraction can be changed counterproductively; simply logging higher/lower levels of abstraction independent of the context or any judgement of its value is not informative.

Finally, as measures of QOC *size*, the number of source Issues per pair was contrasted with the number of merged Issues. A second measure was the size of the merged QOC relative to the mean size of the two source QOCs from which it derived, calculated as:

(no. Issues in the merged QOC ÷ mean no. Issues in the source QOCs) per subject pair

A proportion = 1 represents equal numbers of merged and source Issues, <1 fewer merged Issues, and >1 more merged Issues.

The results are now presented and discussed. Table 5.5 summarises the analysis.

Content of merged QOCs		***Sources of merged QOCs***		***Size of merged QOCs***	
more focussed Issues	7	combined source QOCs	18	total Issues in merged QOCs	**98**
more generic Issues	0	single source QOC	51	total Issues in source QOCs/pair	**108.5**
more focussed Options	0	dual source QOCs	29	mean source Issues /subject pair	8.2
more generic Options	1			mean merged Issues /subject pair	9.0
more focussed Criteria	5			(merged Issues/mean source Issues)/subject pair	0.92
more generic Criteria	1				
SUM improvements	**15**				
(poor/poorer DR	**11**)				

Table 5.5: Summary of the representational changes to independently authored QOCs as they are merged (from scripted-video exercise)

From the source analysis of how the merged QOCs were constructed, of the 98 merged Issues, QOC for 52% was imported directly from a single source QOC, 30% from both, and 18% combined elements from both sources to form a new structure. An example of how elements from source QOCs were combined, is shown in Figure 5.8.

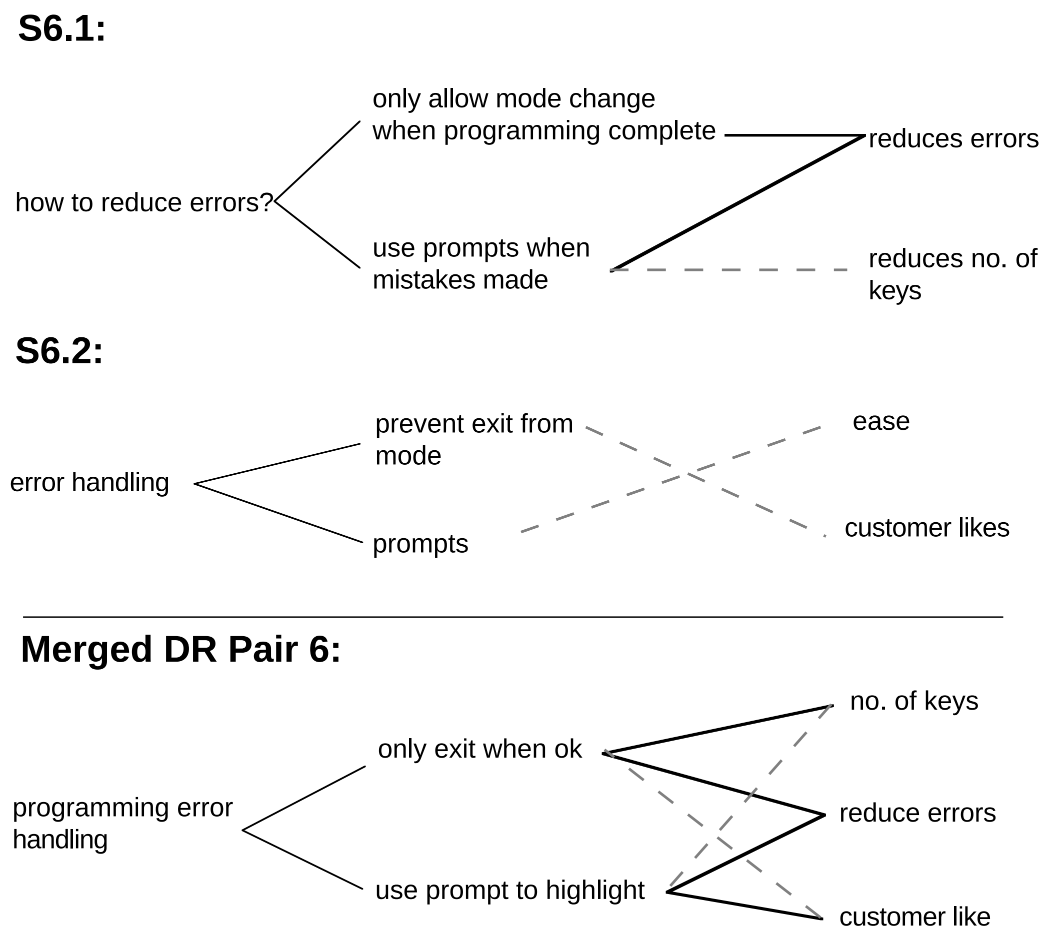


Figure 5.8: An example of combining elements from separate QOCs into the merged QOC.

It is noticeable that just over 80% of the merged QOC was imported without modification (single source+dual source). The fact that nearly a third of source QOC was dual source demonstrates a degree of consistency in authoring, which may be taken as a baseline against which to compare results from future experiments. That is to say, given the same material (the video), about a third of the QOC produced by QOC novices was identical. This figure increases if instances are included where one subject's QOC was similar, but not quite as good as the other's (which was logged as single source).

How can one account for the overriding tendency to import QOC direct from the source QOCs without modification? A critical factor would appear to be that the ideas to be represented as QOC – the design script and sketches – were too well structured. The Issues, Options and Criteria were too obvious to lead to wide differences in representation. If the QOC representations generated individually were sufficient as accounts of the design discussion, transformations would not be needed. This confounded the goal of observing the evolution of QOCs. That being said, whilst this was the case for the majority of the source QOCs, there were also some very poorly organised structures, as illustrated in the next chapter.

In terms of the *size* of the merged QOCs, Table 5.5 shows that they were sometimes several Issues smaller than the average of its source QOCs, and on occasion larger. (Pair 7's data are somewhat irregular, as S7.2 failed to grasp QOC notation properly. This resulted in the S7.1 dominating the session, effectively copying his QOC as the merged QOC, producing a misleading figure for merged Issues). If Pair 7's data are discounted, the proportion of merged to source Issues drops from 0.92 to 0.78, which perhaps reflects more accurately the overhead incurred in the process of merging the QOCs (within the constraints of the allotted 30 minutes).

Lastly, the measures of *transformation* show that overall, changes to QOCs were low in number, six pairs making none at all. Of the changes made, better focussed Issues and Criteria accounted for 12 of the 15 improvements.

As Table 5.5 shows, there were also instances where it was judged that the merged QOC either inherited poor structure from both source QOCs, or the subjects elected to use the poorer representation of the two available to them. Two examples are shown below:

(i) In this example, helpful Criteria in the source QOC are omitted in the merged QOC:

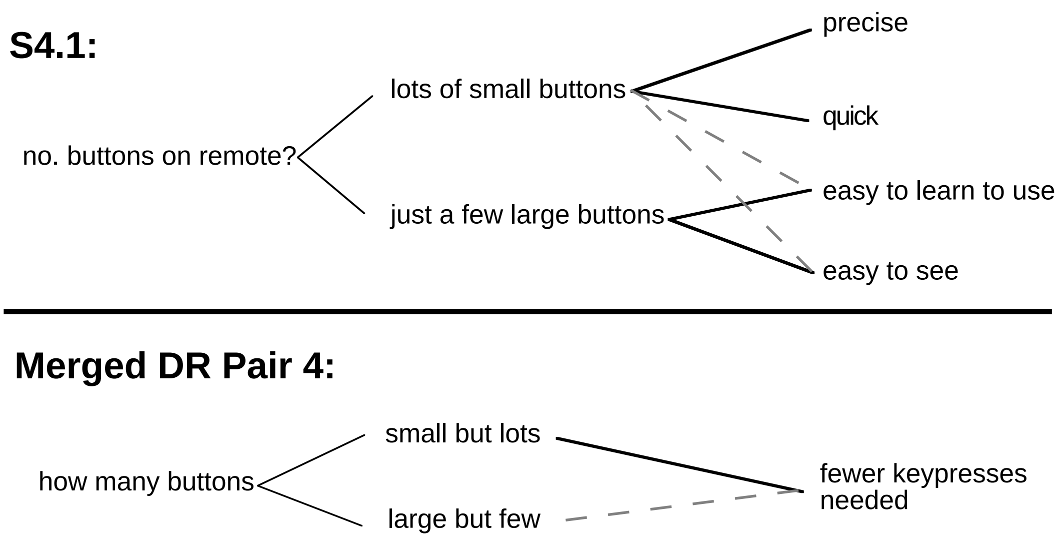


Figure 5.9: Merged QOC with impoverished Criteria compared to its source QOC.

(ii) The two Issues of where and how to display VCR programming information were made explicit by S9.1, but chunked together by S9.2; the merging of the two QOCs led to a single Issue with the two subIssues embedded in it.

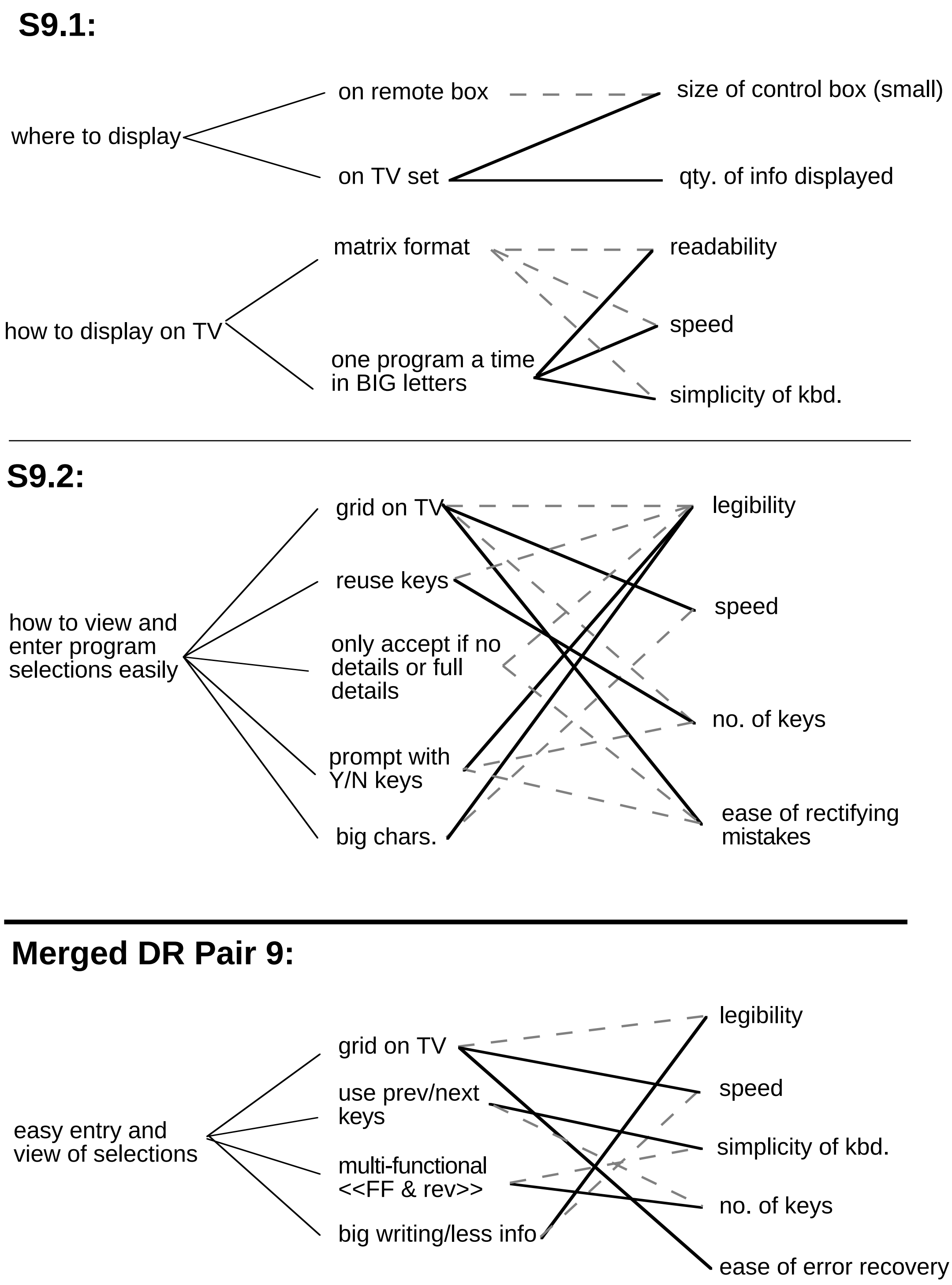


Figure 5.10: Merged QOC with more poorly focussed Issues compared to its source QOC.

The merged QOC in Figure 5.10 illustrates the phenomenon of 'Criterion bunching,' which occurs when Options to different Issues are compared. The Criteria *legibility, speed* and *ease of recovery* assess only the first and fourth Options, whilst the Criteria *simplicity of keyboard* and *no. of keys* assess only the middle two Options. This is a visual cue to designers that an Issue is unfocussed in scope.

To conclude, the results show that the merged QOCs were smaller than the original source QOCs (most likely due to the cognitive overheads of merging, and the limited time available for merging); secondly, the merged representations were found to vary in quality, sometimes poorer, and other times better, than the initial QOCs, with more focussed Issues and Criteria being two of the benefits from merging QOCs. It is not possible to say whether these benefits are due specifically to merging, or simply to reviewing the QOC, as it is likely that any reflection on the state of the QOC will lead to some improvement.

If subjects were given more time one might hypothesise a greater degree of consistency, and if QOC experts performed the same task, the effect might further increase. Increasing the time and expertise increases the likelihood that QOC will be reworked, converging more closely on a notional 'optimal' representation. However, further research is needed to bridge the gap between the data from this training exercise, and the use of QOC by different members of a real design team to represent their own arguments. As already noted, treating highly qualitative data such as DR notations in a quantitative manner is fraught with difficulty. The analyses set out above are best viewed as indicators of the general 'profile of change' which DRs can be expected to follow over time. These are important issues – properties of DR notations (such as consistency, and the effects of different sizes and compositions of design team on the 'response' and 'yield' of the representation) are powerful ways to communicate with the wider software design community how and under what conditions QOC can be used consistently and most productively.

5.3.4 The design exercise: ATM design using QOC

Most of the analyses from the ATM task are presented in detail in the next chapter. Concern will be focussed here specifically on the experimental design of the ATM task conditions. It will be recalled that there were two ATM problem statements [§5.1.3], upstream and downstream, which varied in the extent to which they supplied subjects with candidate Issues, Options and Criteria to consider. The goal was to contrast two situations in the design process: an upstream situation, which represents a state of affairs where there is a limited understanding of the problem due to its novelty, and a downstream situation, in which the designers are relatively familiar with the problem, and have clear in their minds a range of goals to achieve, potential solutions, and constraints to juggle. The hypothesis was that semi-formal notations are only useful in the latter case, and intrusive in the former because of the constraints they impose on the recording of undeveloped ideas.

From analysing the transcripts and QOCs, it became clear that these conditions failed to create sufficient differences in understanding of the problem domain to affect performance with QOC. Any effects which may have been present were swamped by other more powerful factors, namely subject differences in creativity, design expertise, and dynamics within pairs. Limited resources made it impossible to gain access to sufficient numbers of designers who could be controlled along these dimensions.

The conditions themselves could be significantly improved in a future study. The most effective way to increase the level of understanding which downstream subjects have of a design problem, is to allow them to wrestle with it themselves, rather than feed them 'pre-packaged' Issues and Criteria. Thus, it would be in arguing about and designing ATMs themselves that they would have generated a significant mass of familiar ideas to organise as QOC (either memorised or recorded as notes and sketches). Study 2 in fact used this procedure in a limited form by prohibiting the use of QOC for the first 30 minutes for two of the seven subjects. However, primarily due to lack of training in QOC, effects were inconclusive. A revised downstream condition, therefore, would give subjects extra time (30-60 minutes, or even longer) to tackle the problem, before using QOC. In a parametric experiment, between-subjects variation could be controlled by a within-subjects design, such that each pair used QOC for two problems (counter-balanced), one problem in an upstream condition, and the other downstream. This would increase the likelihood that different levels of understanding would exist, helping to reveal more starkly subsequent differences in QOC use, as originally envisaged.

5.3.5 Conclusion: Lessons learnt from developing the tutorial

Several lessons were learnt from this first iteration in the design of QOC training materials. The training exercises (particularly exercise 1) provided the'first glimpse' of designers' use of QOC, which made apparent representational problems and issues which persisted throughout the subsequent studies; consequently, it can be predicted with some confidence that these phenomena will occur in other contexts of QOC use, and probably with other notations like gIBIS or DRL. Analysis of the editing to QOC structures over time for 24 designers resulted in a description of patterns of change which may be generalisable to QOC-use more widely. In this respect, the scripted-video proved to be an effective methodological tool. The importance of choosing the right design problems was highlighted, in order to elicit behaviour of interest. Lastly, it was concluded that the best way to create conditions in which to study design in different phases of development is, essentially, to allow designers to explore the problem in advance, so that they *own* the ideas and knowledge which it is hypothesised will make a difference.

Chapter 6

Study 3: Designing with QOC
Part II: *Main Results and Discussion*

This chapter presents and discusses the main results from Study 3. Where possible, the results have been grouped under headings characterising salient features of the QOC authoring process, although there are inevitably data which bridge classes. The results presented here derive from analysis of training exercises 2 and 3 (printed dialogue on symbol design, and the scripted-video), plus the main ATM design task. Many of the results are illustrated with extracts from the transcripts and QOC structures.

6.1 Analyses of the ATM designs and QOCs

6.1.1 Analysis of the ATM designs (qualitative)

Before dealing with the design rationales, it is important to consider the sorts of design solutions which were generated. Ultimately, proponents of DR who wish to claim that using DR will not impede designers' creative process, and indeed may facilitate design, need to provide convincing evidence. However, it was not the primary aim of this study to evaluate the quality of ATM designs – rather to characterise the cognitive concerns relevant to future QOC authoring environments.

Formally evaluating the 'creativity' of designs is fraught with difficulty. Being such a subjective, amorphous concept – or property – some might argue that the idea of creativity judgements is either arrogant (that somebody feels they can judge creativity), or even impossible. Whilst this view is not held entirely by this author, no attempt will be made to judge the ATM solutions beyond noting novel features to the designs. Work by Thomas et al (1977) exemplifies a more formal approach to evaluating design creativity.

As the ATM problem was underspecified compared to a normal design brief and requirements specification, there was a lot of room for manoeuvre. Designs varied considerably in their innovation, some subjects feeling at liberty to engage in freewheeling conceptual design with a general disregard for cost, others making it a priority that, for instance, minimal hardware changes be made. This inevitably led to different kinds of solutions. Whilst many pairs were able to produce innovative solutions to the problem, the observational methodology used here, coupled with the attendant problems of assessing creativity, made it impossible to make performance comparisons with a 'straight design' condition in which QOC was not used.

An exhaustive log of 'design ideas' would involve analysis at the level of the transcripts, which are a representation of activity at a greater level of detail than QOC. However, the aim was to obtain a moderately detailed characterisation of the content of design discussions, so such a fine-grained analysis was not considered necessary. As a result, the most frequent, as well as the rarer but more creative and novel[1] solutions generated in the ATM design discussions were logged, shown in Table 6.1. Options which were eventually rejected are included, since creativity in design includes as much the space of possibilities considered, as the set of final decisions – for comparing creativity in this respect, the availability of QOCs is extremely useful.

Table 6.1 was constructed primarily from the QOCs, with the addition of Options/Decisions which were recorded in some other form (e.g. notes or sketches of alternatives or the final design). Whilst the QOCs do not represent *everything* that was discussed, it is the case that when a good idea or insight emerged, the designers made an effort to document it. Instances where an idea was recorded, but not as QOC, are signified by pair numbers in parentheses. However, any ideas which were discussed but not recorded in any form are not logged. The Issues have been added to provide some functional context for the Options. In many cases subjects used the same or similar Issues themselves, but where they did not the Option has been logged under the most appropriate Issue.

1 Whilst a *novel* design feature may fulfill its function very poorly, a *creative* feature also works. Whilst creativity in effect implies an element of novelty, the relationship is not reciprocal. However, no attempt has been made in Table 6.1 to differentiate the two.

Issues	***Options***	***Option rejected by Pair No.***	***Decision made by Pair No.***
how many types of ATM?	1 — current cash+services ATM	4,6,8,12	
	1 — cash only FATM	4,6,8,12	(3)
	1 — modal ATM (switches between ATM-FATM)	2	
	many – 1 machine per service	2,7	
	2 — cash only & cash+services	2,10	4,7
	2 — cash only & services-only		7
	1—faster cash + services ATM		**(6,10,11),12**
how to differentiate types of ATM?	only with different keys/display		3
	different colours		**2**
	illuminated signs	**2**	**3**
	no differentiation	3	
how do users identify themselves?	insert card (1 correct way)	9,12	(1,2,3,4,5,6,7, 8,10,11)
	insert card either way up		**9**
	lay card on flat surface reader	**1**	
	swipe card through reader (1 correct orientation)	**6,12**	
	swipe card through reader (any orientation)		**12**
	speech recognition	**1,7,9**	
	fingerprint recognition	**1,7,9**	
	retinal scan	**7**	
	DNA analysis	**9**	
	enter shortened PIN	**10**	
	enter account number	**7**	
what opening sequence of user events?	card → PIN → access services	2,12	(1,3,4,5),6,(7,8 ,10,11)
	access services→card→PIN	**2,6**	**12**
	PIN→access services→card	**6**	
	card→access services→PIN		**9**
how many card slots?	1 slot	5	(1,2,3,4,6,7,8, 9,10,11,12)
	2 slots		**5**
	>2 slots	**5**	
how to select cash?	type in amount	2,8,9,11	1,10
	select one of n-preset amounts	1,8,10	3,7,(11)
	select preset amount or type amount	9	(4,5,6),8,(12)
	select from range of preset amounts set to own preferences, or type in amount		**9**
where to display amounts and services and how to select?	both on screen, selected by entering option number		1,(7,8),10,(11)
	cash on screen, services as buttons	6	(5)
	services on screen, cash as buttons	6	
	all buttons	1	**12**
	both as icons on a touch screen		**9**
	both on screen, selected with joystick/pointer	**9**	
how to select multiple services?	select one at a time	8	(5,10,11)

	select all at once	**12**	**2,8**
	hit any services button anytime		**12**
how to handle deposits?	**user inserts money in and labels envelope**	**9**	
	user enters money & ATM bags and labels it		**9**
	no deposit facility		(1,2,3,4,5,6,7, 8,10,11,12)
should receipt be issued?	yes	9	1,((2,3),4,(5),7, (8),12
	no	4,9	(11)
	ask user	**9**	
	user sets personal preference in advance		**9**
how to print receipt?	print whole receipt from scratch	3	(1,2,3,4,5,7,8, 12)
	add marks to pre-printed amounts & services on receipt		**3**
	2 different types of receipt	**3**	
when to start printing receipt?	**immediately transaction begins**	**3**	
	after checking account		**3**
	last thing in transaction	**3**	
how to issue card, cash, and receipt if any?	different slots	1,2,4,12	(5,7,11)
	all from single drawer		**2,4,(6,8),12**
	receipt with cash, but card separately	9	1,(3)
	user sets personal preferences in advance		**9**
how to handle errors?	clear key	2	
	time out	2	
	block incorrect use	2	
	reduce number of services		11
	speed up feedback on errors		**7**
should repeated attempts at entry be allowed?	**allow first time users repeated attempts**		**4**
	allow repeated attempts	4	
how to indicate termination of transaction?	**explicitly with end button**		**12**
	explicitly by confirming ATM's prompt	**12**	
	implicitly by taking cash	**12**	
how to indicate currently active services?	**lit buttons**	**12**	
	display message	**12**	

Table 6.1: Design issues and alternatives discussed by designers in the ATM task. Novel or creative ideas are in bold. (A number in parentheses indicates that this design pair did not record the decision as QOC).

Creativity in design is reflected not only in the extent of the design space explored (the range of alternatives considered), but in the way in which Options are selected. Following Table 6.1 above, Table 6.2 summarises the Criteria which were most often appealed to by different pairs in making decisions, plus some of the more innovative trade-offs considered. It can be seen that the five key factors, used by nearly every pair, were speed, expense, security (to the bank), number of services offered, and some aspect of ease of use (with examples of its application shown). The five Criteria were also reused to a considerable extent by each pair. The third column shows the extent of Criterion reuse.

Criterion class	***Used by Pair No.***	***Reused at least once by Pair No.***
speed	1-12	1,2,3,4,6,7,9, 11,12
expense	1,2,3,4,5,6,7,9, 10,12	1,2,5,7,9,10
security	1,2,3,4,6,7,9	1,2,6
number of services	1-12	3,4,6,9,11,12
ease of use ***(****e.g. consistency with current ATM, natural ordering, allow keying errors, learnability, fast error feedback, key clutter, concurrency for user input, modelessness, simple attentional requirements)*	1-12	1-12
educational effectiveness *(of leaflets vs. tutorial on how to use ATM)*	**1**	
low paper wastage	**3,4,9**	**3**
useful for personal accounts (receipts)	**4**	
cater for new users	**4**	**4**
software modification only	**4**	
hardware modification only	**4**	**6**
need for user education	**5**	**5**
customer retains card in hand at all times	**6**	
maximise concurrency in the interaction	**12**	**12**

Table 6.2: The most frequently used Criteria in the ATM design rationales. Criteria judged to reflect creativity or novelty in thinking are shown in bold.

6.1.2 Analysis of QOC structures (quantitative)

Let us now consider some quantitative aspects of the QOCs produced in this design session. It is possible to calculate certain summary statistics of QOC structures which are based solely on syntactic patterns, that is, which are independent of any knowledge of the content of nodes. The following measures were used as an overview of QOC use.

Measure	Information provided by measure
Issues	For a given pair, on average how many Issues were there? The number of Issues in an issue-base is a commonly used measure of size.
Options	For a given pair, on average how many Options were there? Whilst one must also look at the *quality* of the Options, the raw number generated is a partial measure of the design team's creativity and dynamism.
Decisions	For a given pair, on average how many Decisions were made?
Criteria	For a given pair, on average how many Criteria were used? Design problems may differ in the size of the Criterion set to which the designers work; the range of Criteria used to make decisions may be a clue to creativity (e.g. decisions always made on the basis of a very limited set of Criteria might be cause for concern).
Decisions without objecting Criteria	For a given pair, on average how many Decisions were made without considering their downsides? (Note: Decisions without *supporting* Criteria do not occur).
Reused Criteria	For a given pair, on average how many Criteria were reused? Whilst the total number of Criteria indicates the range of goals, the reused Criteria focus on those Criteria which formed the 'core set' of goals, i.e. they applied to more than one Issue.
Options without supporting Criteria	For a given pair, on average how many Options (as opposed to Decisions) were rejected without considering their 'good points'? A 'guideline' for representing QOC analyses [§10.3] is that every Option should be assessed both positively and negatively, in order to minimise confirmation bias.
Options without objecting Criteria	For a given pair, on average how many Options were rejected despite having no downsides. Such Options can occur for two reasons: (i) despite having no downsides represented, they do not satisfy the Criteria as fully as the Decision (i.e. relative Assessment weighting [§6.3.4]); (ii) the designers forget to record the negative Assessments (this turned out to be a common omission).
Options with no Criteria	For a given pair, on average how many (rejected) Options were not assessed at all?
Options/Issues	For a given pair, on average how many Options were considered per Issue? Since DSA aims to broaden the space of possibilities which designers consider and represent, this is an important measure when comparing QOC with non-QOC based design.
Criteria/Issues	For a given pair, on average how many Criteria were there per Issue?
Criteria/Options	For a given pair, on average how many Criteria were there per Option?
Reused Criteria/Criteria	For a given pair, on average what proportion of Criteria were reused?
Reused Criteria/Issues	For a given pair, on average how many reused Criteria were there per Issue?
% Decisions without objecting Criteria	For a given pair, on average what proportion of Decisions were made without negative Assessments?
% Issues with 'Yes/No' Options	For a given pair, on average how many Issues simply asked whether to follow a particular Option? (These are not good Issues/Questions in the QOC sense. The aim is to open up the design space with good questions [§10.3).

Table 6.3: Summary of measures used in quantitative analyses of the ATM design QOCs.

The results of the above analyses are shown in Table 6.4 and Figure 6.1.

Measure	Total	Mean/pair designers
Issues	76	6.3
Options	206	17.2
Decisions	73	6.1
Criteria	221	18.4
Decisions without objecting Criteria	37	3.1
Reused Criteria	145	12.1
Options without supporting Criteria	25	2.1
Options without objecting Criteria	15	1.3
Options with no Criteria	7	0.6
Options/Issues	——	2.9
Criteria/Issues	——	3.0
Criteria/Options	——	1.1
Reused Criteria/Criteria	——	0.9
Reused Criteria/Issues	——	1.9
% Decisions without objecting Criteria	——	46%
% Issues with 'Yes/No' Options	——	8%
% Issues with 'Yes/No' Options — *adjusted* *[Only 3 pairs asked Yes/No questions. The adjusted proportion treats them separately from the main data.]*	——	33%

Table 6.4: Results from quantitative analyses of QOCs for the ATM design

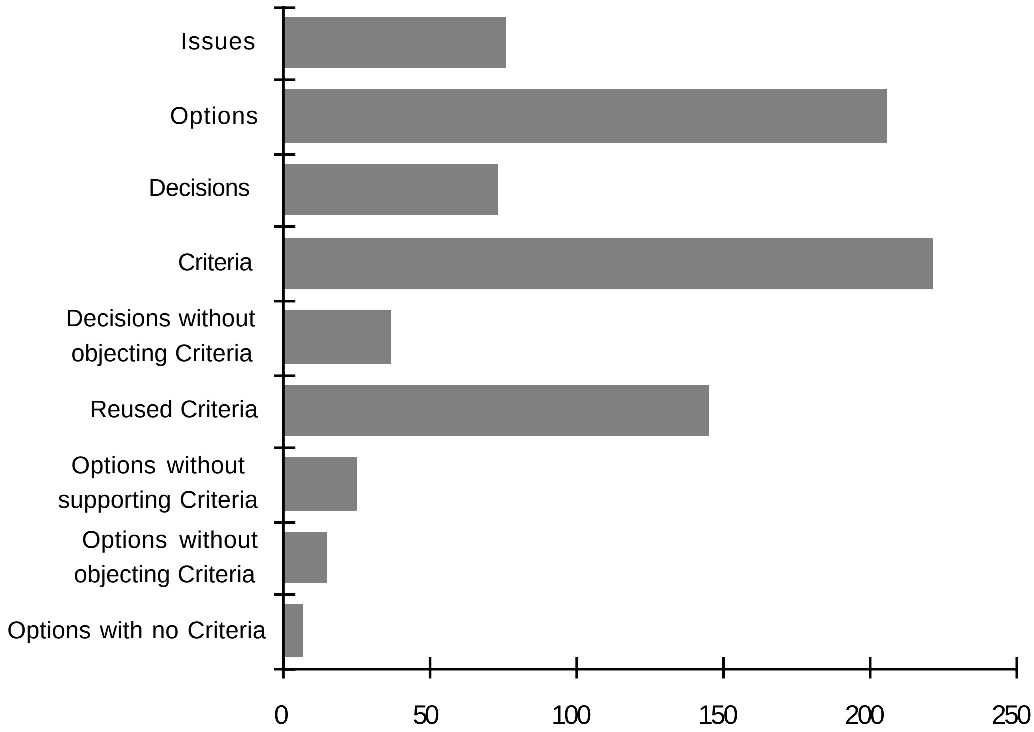


Figure 6.1: Frequency graph from quantitative analyses of QOCs for the ATM design

The results show that the average proportion of decisions made without any negative Assessments was quite high (46%). This would appear to corroborate evidence in cognitive psychology of a tendency to unconsciously seek evidence to support one's own beliefs or preferences (Wason, 1966), and even when faulty to rationalise one's own reasoning as being logical (Evans and Wason, 1976). Adams and Adams (1961), and Lichtenstein et al (1982) also provide evidence of the tendency to ignore evidence contradictory to one's own conclusions. Note however that the actual decisions made have not been individually evaluated, so it is not clear whether they were actually poor. Analysis at a purely structural level is insensitive to such qualitative factors. Secondly, the base level of confirmation bias is not known, which makes it hard to draw strong conclusions. Notwithstanding these caveats, the fact that negative Assessments were not recorded should be of some concern, given that one of the aims of QOC is to encourage exploration of both advantages and disadvantages to Options.

As noted earlier, in order to reduce the informational overheads there was little instruction on the process of using QOC. Greater experience with the approach may be needed before benefits are seen of the sort described above. It is most likely that although the designers were *recording* ideas as DR, they were not using design space analysis to *drive* their discussion as much as they could have. However, detailed analysis of ways of working with QOC is reserved until later [§10.4].

There was a very high proportion of Criterion reuse (mean=0.9, range 0-1.0), with on average just under two reused Criteria per Issue (mean=1.9, range 0.7-2.7). Given that there were on average only three Criteria per Issue, the overall picture is of a problem in which designers used a relatively small set of Criteria (about 18), of which about 12 were reused at least once.

In terms of exploring the *space* around an Issue, decision making on the basis of considering only one Option is clearly undesirable. The incidence of Issues spawning Yes/No Options was relatively low overall (8%). As noted in Table 6.4, however, when the three pairs who were responsible for Yes/No Issues were analysed separately, it becomes clear that such Issues accounted for over a third of all Issues (33%). It is possible that these six designers simply had not learnt to formulate good Issues (principles for'well-structured QOC' [§10.3] was one area given greater emphasis in the revised tutorial developed for Case Study 2).

In summary, this section has provided an overview of the sorts of designs generated in the ATM task. The qualitative analysis shows that together with a number of 'standard' (i.e. recurring) design solutions to the various Issues, the use of QOC did not prevent some pairs from being highly creative in their thinking, generating designs (though incomplete) which displayed novel, workable solutions to the problem. The quantitative analysis demonstrates that certain classes of useful information can be elicited from structural measures of semi-formal representations. This also points to the potential of system-based monitoring of DRs in helping to maintain not only the syntactic integrity of argumentation structures, but also aspects of their quality.

6.2 CORE REPRESENTATIONAL TASKS IN AUTHORING QOC

The user of a DR notation is faced with a series of cognitive tasks as ideas are translated into semi-formal structure. Three basic tasks are deciding what *kind* of an idea one has (classification), how to *label* it meaningfully (naming), and how it *relates* to other ideas (structuring). Before these tasks are illustrated and discussed, however, in order to present a true picture of the authoring process, it is necessary to emphasise the non-linear nature of the above steps, that is, the exploratory, opportunistic nature of the process.

6.2.1 QOC authoring as an opportunistic activity

When observing designers using QOC, it soon becomes clear that externalising ideas as QOC is not a smooth, top-down process. Continual revision and switching from one task to another characterise QOC authoring as an opportunistic mode of working [§1.4], in which the QOC evolves through multiple, sometimes embedded represent-and-evaluate cycles, switching between different parts of the structure.

In the following examples, various approaches adopted by subjects to representing QOC are illustrated, showing how the process of developing QOC analyses is quite different from the orderly structure of the final product.[2]

(i) Pair 2 (ATM): Options, then Criteria are recorded, and the Issue comes last of all:

P Halifax machines drop everything into a little drawer... the Issue here is... well the ideas are AS IS, and EVERYTHING FROM ONE PLACE. The Criteria are...
D what are you going to call the Issue though?
P Hmm, I get caught on the Issues....the Criteria are NATURAL FEEL to it – getting it from different holes doesn't feel natural
D actually, it's more like a teller, more human
P what? If you get it all from the same hole?
D the same kind of thing like when you go the counter, and the guy gives you it through the little slot
P (writes) SECURITY IN MIND (EVERYTHING FROM A DRAW – FEELS SECURE). (Linking to Criteria) – AS IS: it doesn't have a natural feel to it, everything from different slots.
What's the Issue here? (frustrated tone).
D erm... I suppose physical layout of...
P layout of holes (starts to write)
D physical layout of input/output stuff (P writes LAYOUT OF I/O FOR CASH/CARD/RECEIPT)

(ii) Pair 10 (VIDEO): Figure 6.2 shows the order in which the QOC was constructed, illustrating switching between Issues to get a new idea down.

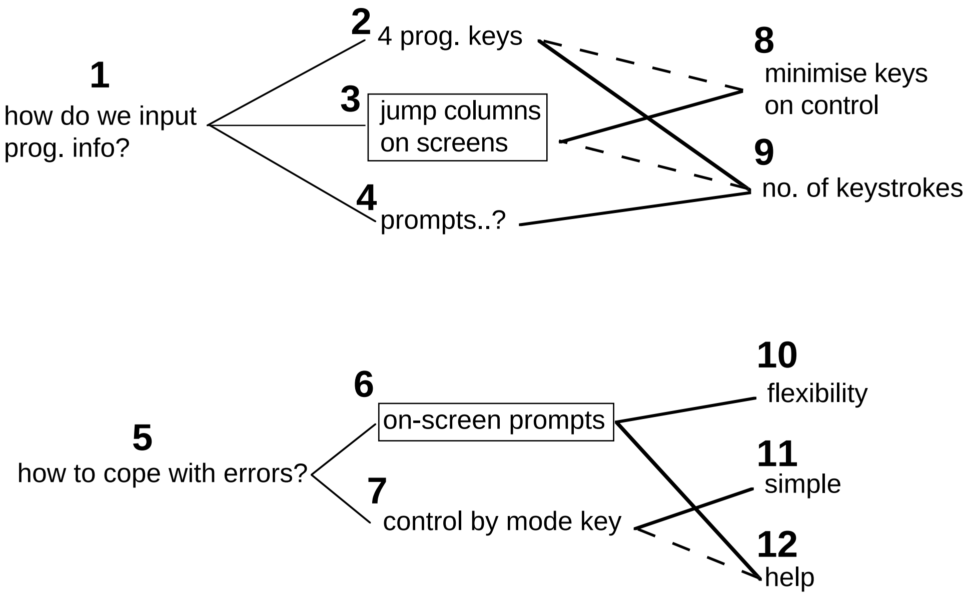


Figure 6.2: Switching between Issues in representing QOC.

(iii) Pair 12 (ATM): In some cases, subjects explicitly adopted 'strategies' to representing the QOC, as ways of imposing some structure on their task (in a sense, exercising control over the tendency to pursue new ideas as they arose opportunistically). Pair 12 adopted the following strategy:

[2] In transcript extracts, the key points are boxed; upper case text refers to the names of QOC nodes .

G	let's try to write down some of the Issues of concern here, and then some Criteria, and then I think some of the Options will come from that – possibly... 'cos Issues and Criteria are related a lot aren't they
J	so if we have one heading Issues and another here for Criteria, which *might* not be related
G	right – I'm going to concentrate in terms of Issues, and then the Criteria might come from the Issues. LENGTH OF QUEUE is an Issue. USER ERROR...

About a minute later, they found themselves beginning to discuss the details of a possible Option but 'held off' intentionally, aware that for this early stage they had decided simply to record as many ideas as possible:

G	so could you have a machine that behaves like a regular ATM or a fast ATM depending on where you put the card in?
J	you *could*... but would that be ... right, so
G	yeah – so (writes) VARIETY OF MACHINES ... (no further discussion on that Option until later)

It is important to note, however, that strategies of this sort seemed to be short-term, flexible modes of working, that is, they did not govern the structure of the whole session, and within them there was flexibility to attend to different parts of the QOC, and to switch strategies, such as:

- listing Issues in advance, before elaborating them;
- generating Options and Criteria, and then the Issue;
- generating Issues and Options first, and then evaluating with Criteria.

QOC authoring is not only opportunistic *during* conceptual problem solving of the sort in the ATM task. Even when the decisions have been made and all the associated rationale is known, working out how best to *represent* it is a separate task (although made easier after the event); indeed some of the examples in this chapter are taken from the training exercises, for which subjects were not generating any of the material to be represented.

Having recognised that the authoring process is 'messy' as opposed to a tidily sequenced activity, let us now turn to its constituent tasks.

6.2.2 Learning to classifying ideas

This section focusses on the normal process of classifying ideas. When QOC is being used, the content of discussion always involves reference to the notation as the QOC is developed (*that's an Option; could that be a Question?; this Criterion keeps coming up,* and so forth). As with any language, fluency increases with use, and arguments are more appropriately classified. In this study, it was found that often, subjects classified ideas without spending much time discussing what type they should be. However, as the examples of poorly QOC show [e.g. §6.2.3, §6.2.5], the first translation which springs to mind is not necessarily the optimal representation.

It is notable that even though compared to novices, 'experts' with QOC (e.g. QOC's developers) are able to develop well-structured design space analyses more quickly, they still engage in restructuring, reclassification, and renaming; these revision activities are dealt with shortly.

Some difficulties which arose involved more complex representational problems in which it was difficult to use QOC; these are dealt with as expressiveness issues [§6.3]. However, others were most likely due to inexperience in using QOC. Three incidents exemplifying translation problems typical of relatively novice use of QOC are shown below.

(i) Pair 2 (ATM): A typical error in classifying an idea (Option instead of Criterion):

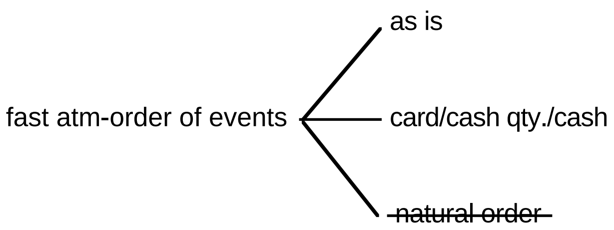


Figure 6.3: A typical error in classifying an idea (Criterion as Option).

(ii) Pair 2 (ATM): an Option is initially represented as an Issue:

D what does this do then? ...the Fast ATM do, if you press the cash amount and not the card?
P I guess it just goes Ha Ha, and clears itself. It'll have a TIME OUT, or CLEAR KEY.
D yeah, but that's an Option though – a Clear key.
P on where?
D well on the normal ATM. You could change the order of the events, but you have a clear key and a timeout

P FAST ATM ORDER OF EVENTS (writes as a new *Issue*)... that's an Option?
D yeah that's an Option
P oh, it's an Option which addresses this security issue isn't it (deletes Issue: CLEAR KEY AND TIMEOUT ON ATM, and writes CLEAR KEY + TIMEOUT as an Option. What's the Issue? It's sort of a vandalism issue

(iii) Pair 5 (ATM): This extract shows the pair generating ideas, and then striving to represent those ideas as QOC:

R that's a design decision (suddenly pointing to the keypad sketch) – deciding that Cancel is going to be the emergency getout, and that we should stick with that as its mode – the mode of that key is 'get out of this, now'.

J well it certainly is... it goes back to this question of BUTTONS – is that what the Cancel... I mean we've added the Cancel key, and consider that to be a design conclusion – I like that because it's intuitive. I suppose that could be a Criterion – in fact it's a very important user-Criterion. (sigh) How do we record that?

R like the arrangement of numbers: the reason I wrote them like that is because it's a telephone pattern (i.e. a familiar layout)

J well ok, that's fair enough. We've not got much time left. We could do with identifying this Cancel key more specifically. But I don't know what the Issue is.
R in fact the Issue is 'how does the card get returned?'

J you can put it under the question of BUTTONS, because you've actually got... you're allowed to have – and I'm being pedantic here in terms of the design rules of the game – but you can actually have multiple boxes down here (i.e. selected Options), and I think the Enter key and Cancel key are part of the question of BUTTONS. That's the Issue (Q. OF BUTTONS) — now there are probably other Issues which feed into this, and I don't know that you can actually... Hah! (laughs) When is something an Issue, and when is it a Criterion?

In the above extract, the decisions and arguments are clear in J.'s mind, but neither he nor his partner are sufficiently fluent with QOC to translate them; in this stage, encoding QOC is difficult, and more of a barrier than a tool.

It is concluded from these and the other incidents logged under 'classification difficulties,' that designers learning to use any of the current DR notations will encounter this representational gulf between concepts and their DR translations. Judging the appropriate size of vocabulary is a difficult task: one walks the tightrope between offering an impoverished range of types on the one hand, and a bewildering range on the other. This problem and possible solutions are discussed in more detail elsewhere [§2.2.2, §10.6].

6.2.3 Naming and renaming

As the section heading suggests, naming is often a process of renaming. The renaming of nodes was a prevalent activity, in which every subject pair engaged. Renaming reflects the problem solving *process* of developing ideas; if a QOC is constructed *as* the problem is explored, it is inevitable that node-names which do not reflect current understanding of the problem must be updated.

Naming takes up a significant amount of time in QOC authoring for several reasons. A node's name must be succinct, and convey the idea it represents. This is the case with graph-based structures of the sort currently used to represent the QOC, or other representations, such as a matrix of Options against Criteria. To aid interpretation, a constraint on Criterion names is that they should be expressed positively, e.g. *easy to learn*, *low error rate*, *low cost*, *high speed*, or at the very least, neutrally, such as *speed*, *cost*, *clarity*. Under this constraint, *supports* Assessment links always signify 'pros', and *objects-to* links 'cons' for that Option.[3]

One important property of names is *focus*. Focus refers to the level of generality at which the idea is expressed: a Question may address several issues; an Option may embody several key features which differentiate it from others, but not along the dimension which is addressed by the Issue; a Criterion may be expressed so generally (e.g. 'usability') that it is hard to see how it relates to an Option. Not entirely independent of focus, the other important property of a name is its relationship to others of its type: it should be

3 Because Criteria also have different weights [§6.3.3], decisions cannot be made purely on the basis of how many *supports* links Options have, but this provides an initial visual indication.

distinctive. An Option may really be an example of another; two Criteria may express a trade-off; two Criteria might really be saying the same thing in different ways. Both distinctiveness and focus in naming are characteristics of 'well-formed' QOC [§10.3].

Although these requirements were not made explicit to subjects, the following examples of the naming process indicate that they appreciated the need to find good names for ideas. In general, subjects were able to devise a meaningful name for their idea without excessive trouble (i.e. without disrupting the flow of conversation for more than a few seconds). The examples illustrate the cooperative process of refining names, as well as incidents of poor naming, due to being over cryptic (see also §4.2.5.1), or too unfocussed; this made them meaningful only to somebody privy to the deliberation context in which the idea had arisen. (All the examples of structuring QOC later on [§6.2.5] also illustrate renaming, although specifically in the context of Issues).

(i) Pair 4 (VIDEO): In each of these two extracts, the designers work towards a succinct Criterion:

R so how are we going to...
T keys to what kind of functions...
R that's not a very good way of putting it...
T it's like the classes of functions...
R classes! That's the way to put it.
T WHAT CLASSES OF FUNCTION KEYS.

R you see teletext is the only thing you read – you don't read other things – you don't read the picture.
T what we have are two negative reasons
R we have to make them positive though...ok, so easy to read?
T well, that wasn't the point was it? um... it was like that they couldn't actually...
R they can't see it, so they don't need it.
T (laughs) yeah -- it's like the Criterion is that you're providing a function which they can actually make use of, and they can't make use of the teletext because it's too small to read.
R ok – USEFUL FUNCTION?
T yeah, ok.

(ii) Pair 8 (SIGN): A Criterion MISLEADING is changed because it is too unfocussed (it is also negative):

T so why didn't they used that one? — (writes) MISLEADING. So the STOPWATCH isn't MISLEADING, but that one is (HIGHLIGHT 1). So you've made a decision there on the STOPWATCH haven't you? But then we've got another question, Issue here.
J STOPWATCH WITH 1 AT TOP's MISLEADING isn't it?
T Just the STOPWATCH doesn't show time passing... Hang on (returns to dialogue) So why was the clock with highlighted 1 rejected? – it was showing too much information
J maybe we should change MISLEADING, 'cos that covers a lot of things – that covers 'seems to be telling the time...'
T (deletes MISLEADING) There's a Criterion there for not choosing that one and for looking at the STOPWATCH again. Is that because...
J it seems to be showing the time, rather than passing of time
T so Criterion (adds in place of MISLEADING) NEED T... oh hang on – (consults dialogue) here we've got "no extra information". So I think that's rejected on the fact that it shows too much information...

(iii) Pair 12 (ATM): This example shows how two Criteria which arose naturally in discussion needed to be re-expressed positively in QOC:

G ok, so we're going to design one machine
J so the FAST is what they've got – Criteria: FUNCTIONALITY
G and also CUSTOMER PREFERENCE.. which ties in with FUNCTIONALITY. SPEED
J and usability in terms of confusing you – USER CONFUSION
G the FAST one's going to be yes for the SPEED... that's not phrased in the right way (USER CONFUSION) 'cos that sounds negative
J (changes to) CLARITY
.....
G questions of error as well
J ERROR .. Are these both negative on SPEED? (STANDARD and MULTIPURPOSE)
G yeah. The MULTIPURPOSE is.. there's a ranking there 'cos it could be fast – it's a question of whether
J (links in neg. links) it's then an Issue to see how we can then improve the SPEED
G has MULTIPURPOSE got a link to CLARITY
J yeah
G (re. ERROR) FAST I would say is.. say ERROR FREE (changes Criterion)...

(iii) Pair 5: This quote from debriefing describes the experiences of many subjects in naming Criteria positively:

R ... I mean, I really struggled on that first exercise, and found that very awkward and very difficult. In fact the thing I found most difficult was negating everything, so that the attribute was a *positive* attribute
J yes
R I just couldn't get my brain to pick out the right word to describe that attribute.

Having demonstrated the cooperative process of naming and renaming, let us now consider the task of structuring the QOC.

6.2.4 Structuring QOC: Asking the right questions

As described in the introduction to QOC [§2.1.5], QOC structures are organised by Questions (or as in this study, Issues). The primary structure which presents itself to somebody browsing the QOC is the Issue structure, and it is under Issues which all design ideas must be eventually placed. For this reason, the problems which designers choose to address through the Issues are important; the problems they pinpoint define the space which the team sees their design occupying, and guides the direction of future deliberation. Many examples of Issue formulation and reformulation were collated in this study. Renaming of Issues is dealt with here rather than 'naming,' because of their importance in defining the macro-organisation of the QOC.

(i) Pair 12 (ATM): Working out the Issue:

G	(new sheet) Our first Issue – do you want different kinds of ATMs for the different – you know, a fast ATM and a fully functional one – or do you want to do it all at once?
J	ok, so what's the Issue – 'cos those are the Options aren't they?
G	do you want..um.. well the Issue is...em...
J	single machine
G	yeah, kinds of ... do you want to have just one machine or do you want to have...
J	well those are Options
G	yeah, well I know those are Options (laughs), but the Issue can kind of beg the question
J	well it could be an Issue – do you want a variety of machines? Yes or No
G	well NUMBER OF ATM DESIGNS: one and more than one – typically two: fast and fully functional

(ii) Pair 10 (ATM): The subjects return to their first Issue, and realise that it no longer expresses what they have now identified as the real problem (the design of the first screen):

T	oh no. (pause – returns to Issue) Except that this isn't really how to develop user interface – it's the first screen isn't it?
A	what do we show initially?
T	yeah
A	(changes first Issue) WHAT DO WE DISPLAY ON 1ST SCREEN?

(iii) Pair 2 (SIGN): In this extract there are several revisions:

P	(reading) should we have any people in it? ...this is to do with communicating that luggage is left and secure (points to Issue), and we introduced the attendant.
D	that's our Criteria there (points to the Issue statement) – that it's left and secure.
P	are you saying that's a Criterion? (points to the Issue)
D	no, the Issue is
P	you can put the Criterion that it is 'left' luggage, here (as a Criterion). Does putting a person imply that it's left?
P	if you put 'left luggage' and 'secure' here (indicates Criteria column)... I think we've got two issues here (points to Issue statement)
D	that's what I was just thinking
P	this is security, and this, and this (indicates all of them),- nothing to do with it being left luggage ... (D deletes 'left' part of Issue statement)
D	(adds 'left' to first Issue after agreement)
	(continue to address Criteria) should we have clarity (ie. semantic mapping) and simplicity (ie. visual clutter), or are they the same? Change CLARITY to 'clear meaning'.

(iv) Pair 4 (SIGN): Having represented two Issues, it becomes clear that one derives from the other, leading to a more general naming of the first Issue:

R	(pointing to Issue HOW TO REP. PASSAGE OF 1 HR?) Perhaps we should change this to 'how to represent *time*?'
T	yeah ok.
R	and then the next one to how to represent passage of 1 hr? (T. changes the Issue). Because this is just the general... (indicates the current Options to the Issue).
T	there's almost two steps of refinement, because next you have "how do you represent an analogue clock", and then (vague gesture, probably indicating the various analogue designs discussed in the script).

(v) Pair 2 (ATM): An Issue is refocussed as it is realised that the Options serve a common purpose (user education):

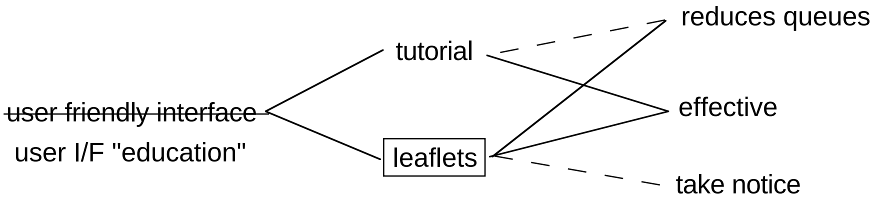


Figure 6.4: Refining an Issue in response to its Options

D Issue: USER FRIENDLY INTERFACE (P writes this). An Option is a tutorial, but at peak times that's going to be useless – someone sitting there and reading off all this stuff, that's going to clog the thing up completely; get two or three of them and there's your queue.
P (writes Option TUTORIAL). (working out Criteria) so that's got to be...
D it's got to be a negative association – REDUCES QUEUES, i.e. it doesn't reduce queues.
P are people really going to be bothered to stand up at the cash machine to learn how to use it?
D well, I don't mean idiot proof; it just prompts you through it, slowly. And if you do something wrong it says 'uh oh' and alarm bells ring, that sort of thing – there's an idea – put alarm bells in!
P USER FRIENDLY INTERFACE: we're going to have a tutorial... no this isn't USER FRIENDLY INTERFACE, this is USER INTERFACE EDUCATION (changes). LEAFLET (adds new Option). You see you can effectively do the same thing with leaflets as with tutorials – a step by step going through, so if anyone's really hasn't got a clue they can take the leaflet up to the machine and it'll take them through it. (links in to Criterion REDUCES QUEUES; adds Criterion EFFECTIVE).

As Bellotti et al (1991) emphasise, asking the right Questions is critical to developing a useful design space representation, and avoiding particular mental sets or design fixations (Jansson and Smith, 1991). The data collected in these studies in fact demonstrate that Issue/Question revision is to some extent a process in which designers naturally engage when using a DR notation, even though the designers were not explicitly told to refine Issues. Design space analysis attempts to build on and support this crucial activity.

6.2.5 Restructuring QOC

'Restructuring' in this context means the *transformation* of structure, by changing the organisation of existing QOC (such as moving Options between Issues, or making criteria which are implicit in an Issue explicit as Criteria). Note that whilst the addition–transformation distinction is useful as a task analysis, the two often occur together during authoring.

It has been described how the reformulation of Issues often takes place in reaction to the Options generated. The relationship is reciprocal, however: an insight into the nature of an Issue can lead to restructuring of those Options, by moving them to new or existing Issues. Examples of restructuring are presented below.

(i) Pair 8 (SIGN): Options to an Issue are separated as the Issue is decomposed into two Issues (T. separates the Options to indicate that STOPWATCH and subsequent Options now respond to a second Issue):

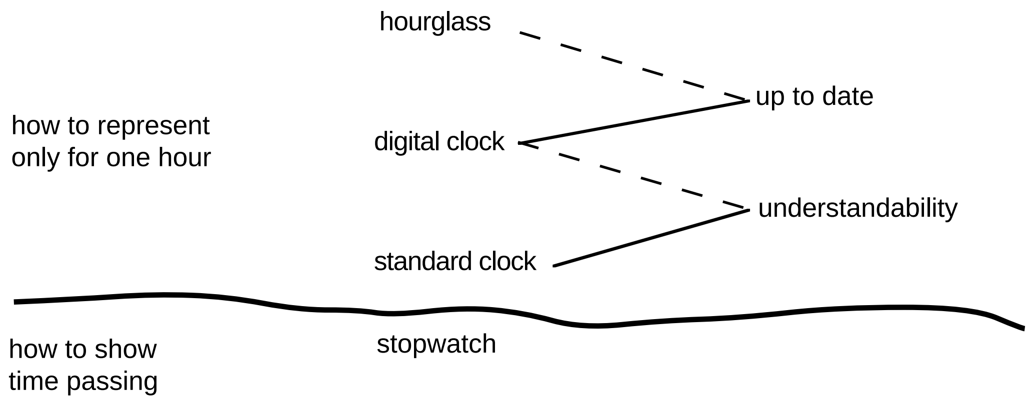


Figure 6.5: Dividing an Issue into two as it is realised that the Options serve two different roles.

(ii) Pair 8 (SIGN): A misunderstanding between subjects has led them to represent the same Option in two different ways. When they realise this, they decide to merge the two into one, but find that the Options do not have the same Assessments:

T (stops short) oh I see – so the BAG IN SAFE and the BAG IN 3D is the same thing.
J yeah.
T right. So what have we got going out of BAG IN 3D DRAWING? (i.e. what Assessments?) It's REALISTIC, but we've said the BAG IN SAFE isn't REALISTIC...
J (reading dialogue) "the three dimensional drawing's getting really complicated, though it looks more realistic."
T oh right – so that's wrong (lAssessment from BAG IN 3D DRAWING to REALISTIC)
J it's that one that's wrong – the dotted one (from 3D DRAWING to REALISTIC).
T so if we scrub that one out (deletes BAG IN 3D DRAWING) we've got two Options...

(iii) Pair 10 (ATM): This longer extract shows how several ideas were initially recorded as Options, but the clue that they are better represented as Criteria is that the subjects want to 'do' all of them – they are *goals*:

A yeah, one Option is to REDUCE THE RESPONSE TIME; reduce... MINIMISE KEY DEPRESSIONS
T yeah
A minimise number of screens
T do you think that's mebbe a Criterion? (A. laughs – seem to recognise this problem from before) They are I think – they're Criteria (long pause while they contemplate the QOC)
A if the Criterion's reduce response time, what's the Option that's going to do that? – invest in CPU, or hardware...
T ugg.
A the other one I was going to put is minimise no. of screens
T yeah
A I'll put it down for now. (as an Option) MINIMISE NO. OF SCREENS
T I feel like we're going at this in completely the wrong way, but I can't think what we should do (reads some more of problem statement)
A I mean really for any system we're developing, we would want to do all of those three (Options) wouldn't we? When you say you've got to choose one course of action ... that's the way we've done it before
T that's why I think they're the Criteria
A yeah – really we want to pick all of those if we can
T that's why I think they're here (points to space left for Criteria) – those are the things we're trying to achieve aren't they? So we'd have something that says, I don't know... (trying to think of Options)
A just re-develop user interface
T yes, re-develop same screens – make them look nice – only that wouldn't be any good 'cos it's not reducing the NUMBER OF SCREENS, or something like that. So they're our Criteria
A (adds Option) REDEVELOP USER INTERFACE

. . . . .

C	are we agreed these are three Criteria?
A	yeah (deletes them as Options and adds as Criteria; they go on to generate Options and evaluate them against their new Criteria)

(iv) Pair 8 (ATM): Subjects make the Criterion INCREASED QUEUEING[4] explicit, rather than leaving it embedded in the Issue. It can then be used to differentiate between the two Options:

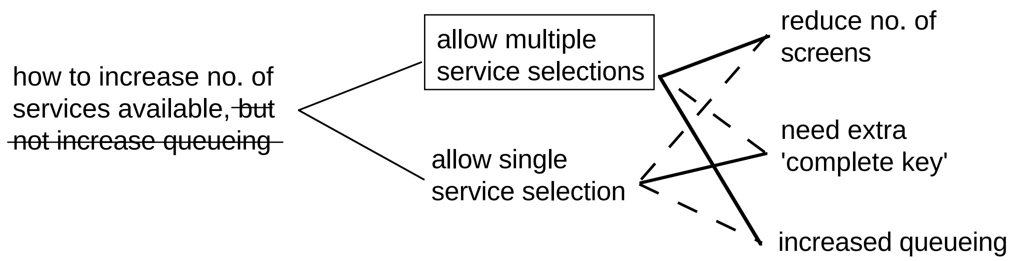


Figure 6.6: Extracting an implicit Criterion from an Issue, and making it explicit as a Criterion.

J wait a minute – we could have ‘how to increase number of services available’ (as an Issue), and have ‘increased queueing’ as a Criterion
T um, so what would we have?...
J you could have press all service buttons required, and then commit, or press each service as required, and as one of the Criteria have queuing (long pause)
T so one was ALLOW MULTIPLE SERVICE SELECTION or
J allow each service

T	oh – (deletes BUT NOT INCREASING QUEUEING from the Issue, and adds Option) ALLOW SINGLE SERVICE SELECTION. If you allow multiple service selection you need an extra key. (Criterion) NEED EXTRA ‘COMPLETE KEY’ (links in). (Criterion) INCREASED QUEUEING (links in).

The above sections illustrate how QOC names and structure are continually modified as ideas develop. Two conclusions are drawn from the prevalence of revision activity which occurs in QOC authoring.

(i) What marks QOC out from other approaches to capturing and representing DR is its emphasis on *reflecting* on deliberation, in order to map out a carefully considered space of possibilities plus argumentation (atemporal, logical DR), rather than record arguments in the form and order in which they initially occur (process driven, chronological DR). Although it is impossible to represent the complete design space around an artifact, understanding of the space develops with experience, over a given design’s development. The form of the space is clearer at the end of a project/project phase than at the start. Consequently, revision of names and structure is a natural part of the QOC philosophy, because rationalisation means revision.

(ii) Be that as it may, revision is surely not confined just to design space analysis. Although there are no detailed analyses of DRL or gIBIS in use, it is highly unlikely that

4 This is also another example of the difficulty subjects had in remembering to phrase Criteria as positive attributes.

the modes of working described above are unique to QOC. Indeed, it could be argued that given the limited QOC expertise possessed by subjects in the studies reported in this thesis, their task was as close to gIBIS authoring as design space analysis. Subjects did not have sufficient training or experience to be able to appreciate in practice the difference between process versus structure oriented DR.[5] The restructuring activities which took place generally involved reorganisation of ideas within a local context, rather than resulting in completely different perspectives across large sections of the DR. Whilst design space analysis certainly includes expanding 'microspaces' around specific aspects of the design, does not gIBIS and DRL authoring involve very similar activities?

In gIBIS for instance, it is likely that an Argument which embodies two subarguments might be decomposed as such, to make those elements explicit, especially if they related differently to different Positions. Likewise, gIBIS Issues (which are functionally equivalent to the Issues used in Study A) are likely to undergo renaming and refinement to some extent, although the approach does not place so much emphasis on conceptual organisation as QOC. Similarly, in DRL one would not expect ideas to be encoded optimally first time, every time. The envisaged mode of use of DRL is as a reflective tool, not for use during meetings. Whilst providing more time to plan the representation, this also opens up more scope for restructuring than gIBIS. One can foresee the need for certain DRL transformations which would only become apparent as ideas were explored. For example, one might wish to transform a Question into a Decision-Problem, if it opened up a new, previously unexplored argument space, and became important enough to be a Goal. DR tools should be able to ease the translation from one type to another [see §10.7].

In sum, revision is inherent to DR authoring, by virtue of the fact that design problems are 'wicked' (Rittel and Webber, 1973), and are rendered manageable only through the exploratory process of framing and reframing the problem. Further discussion of similarities between narrative and retrospective DR authoring can be found in §10.8.

6.3 EXPRESSIVENESS AND REPRESENTATIONAL FACILITY

It will be recalled that expressiveness in a representational scheme describes the coverage of the domain – can all of the important concepts be expressed in the representation's vocabulary? This section reports several areas in which QOC notation, as it currently stands, was judged to be inexpressive.

[5] Several designers did however comment that it is impossible to rationalise ideas immediately; it is necessary to reflect on what happened after the event.

6.3.1 Inter-Option relationships

Several relationships between Options arose in the course of this study, falling into two classes: Option relationships within Issues, and those between Issues. Each is illustrated with a summary of the context in which the problem arose, and implications for QOC considered.

6.3.1.1 Option evolution: The problem

An important point, highlighted initially by the graphic design training exercise, is the question of how to represent the evolution of an Option over time, as it is refined. In the design dialogue in this exercise, the basic design underwent a series of changes as first one, and then other ways of representing "1 hour passing" were proposed. Subjects encountered some difficulty in representing evolution as QOC. One member of Pair 10 observed:

T generally speaking, those ones [Options] sort of meet our requirements, but in increasing amounts [moving from 2nd to 4th Options].

A member of Pair 4 summarised the problem as follows:

R This is more a development of ideas though than a load of choices; you know, each one's better than the last.
T yes, almost.
R rather than a 'well they're all equally valid.' It's more like a... if you could have a row of things, and say 'this is what I started with, and then I added some extra bits, and this was this..' and it's more a logical progression than a choice between things.
T yeah – I don't think we've done very well as regards breaking down the decisions.

Not surprisingly, the evolutionary development of ideas is by no means unique to graphic design, although the medium makes heavy use of combining different elements. In Case Study 1, the designer commented on this precise matter in relation to the way in which he was working:

> ... it's more a case of you don't really see the other (Option)... until you test it. At this stage you see that (indicates an Option), and it's all you can think of doing – the first Option if you like. And it's not an either/or situation.. this *was* a case where there were decisions to be made, but most of the time it's more subtle than that... there's one thing made up of lots of interlocking bits, and you only find out what's wrong by testing it. You do have Criteria of course – it's got to enable certain flexibilities and possibilities to happen, but you don't then say 'well that one was negative for that, so we won't do that protocol, what's this other one?' It's really that the second Option only developed out of trying the first one, and may be based quite a lot on the first one as well.

Note the similarity with previous comments. The *process* of design at the points when these comments were made was better characterised as one of refinement of a single Option, rather than the selection of distinct alternatives. The problem seems to be that QOC (and DR notations in general) are based on a model for representing competing alternatives to a given problem, rather than the evolution of a single alternative over time.

It might be argued that the use of QOC to represent the evolution of Options over time is in fact counter to the approach of design space analysis, which aims to communicate a static view of the space in which a design sits, rather than 'how it got there.' The emphasis should be on representing the 'corners' or extremes of the space as contrasting Options, rather than detail local regions of the space. One might further argue that a complete design should not be represented as a single Option – rather, it should be broken down for analysis by Questions which address its defining characteristics. Unless it is extremely simple, a design consists of a set of defining features (Questions) and associated Options, not one Option. According to this view, the evolution of a design extends the design space in different directions through the addition of (i) new Options as new ways of resolving a problem are generated, (ii) new Questions as new 'dimensions' to the space are discovered, and (iii) new Criteria to show how the new versions of the artifact improve the old ones, and the trade-offs incurred.

However, even if *completed* QOC analyses in general factor out small refinements in ideas, the issue remains of how to manage idea evolution whilst 'design is still happening.' What is missing is a view of how QOC will support the process *through which* well-structured DRs should emerge. Certain kinds of design seem to proceed in an exploratory fashion, during which Options evolve as weaknesses are uncovered. Even if in the final analysis the particular path of evolution is of less concern than the distinctive alternatives, it is a requirement that DR notations intended to support authoring should take the representation of Option refinement seriously. Possible ways to do this within the philosophy of design space analysis are considered next.

6.3.1.2 Expressing Option evolution in QOC

Let us start by considering a 'pseudo-QOC' representation generated by a pair of subjects in this study. The evaluation of their scheme raises representational issues which need to be resolved by other schemes.

Pair 8 unintentionally[6] produced a QOC structure which conveyed process information about how an Option had evolved (reproduced in Figure 6.7). This QOC shows how each Option was rejected in favour of a better version, starting from the top Option. Each Criterion serves only to object to the 'current' Option in order to show the next step of refinement. For instance, the third Option, STOPWATCH WITH 1 AT TOP was devised because STOPWATCH failed to SHOW TIME PASSING; however, because it was judged MISLEADING, STOPWATCH WITH SOLID LINE POINTING TO 1 was devised, and so forth.

---

6 There is no indication that they were aware that their structure displayed the properties discussed here.

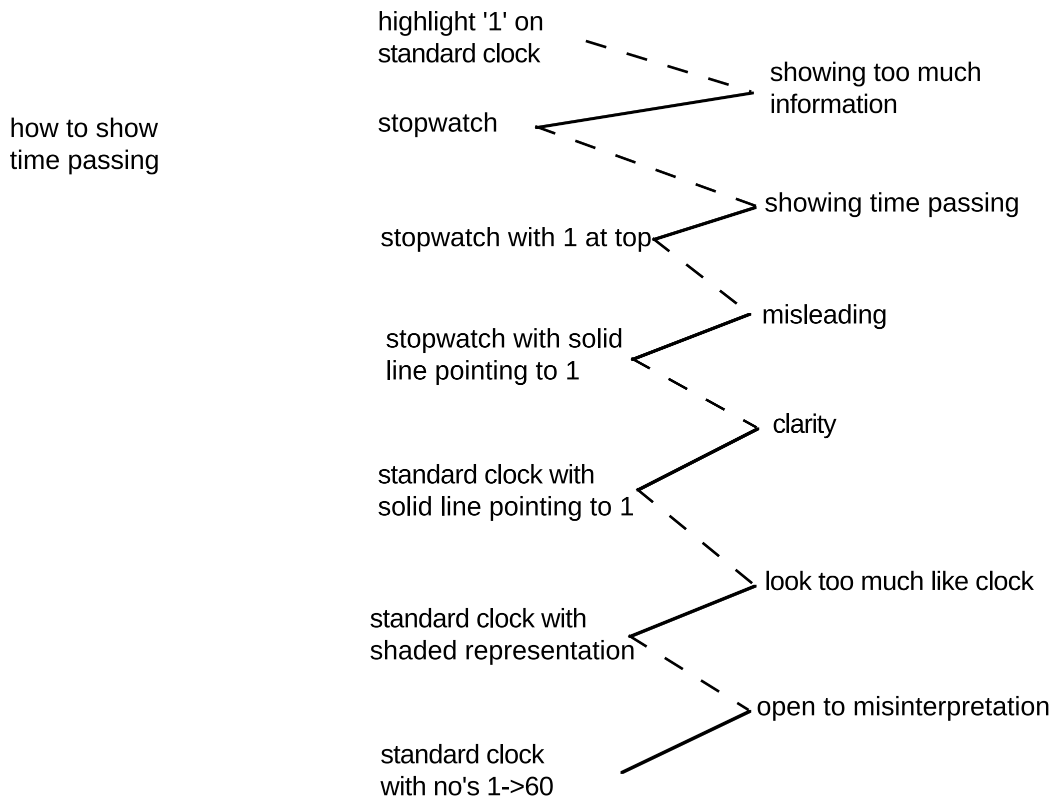


Figure 6.7: One way to capture the evolution of Options within QOC.

Whilst this is an interesting way to incorporate process information into QOC, it is not a good representation on several counts: (i) all but one of the Criteria are negatively expressed; (ii) Criteria like OPEN TO MISINTERPRETATION and CLARITY are too general to be able to understand their application to the Options; (iii) in well-structured QOC, a Criterion is linked to every relevant Option, acting as a comparative means of evaluation, whereas in the above structure, however, a Criterion is only linked to two Options to show the main reason why a modification was made. This leads to two ambiguities:

- The absence of Assessment links to other Options implies that the Criterion is irrelevant, which in the above example is clearly not the case (e.g. SHOWING TIME PASSING applies to all Options)
- Using an *objects-to* link to an Option to show why it was rejected fails to communicate that the Option may in fact satisfy that Criterion more fully than its predecessors.

It can also be seen that the Criteria MISLEADING, CLARITY, and OPEN TO MISINTERPRETATION are essentially the same, which breaks the QOC convention of not repeating Criteria within an Issue [§10.3]. In other words, whilst using the same entities (Q, O, C), the way in which they are used violates the notation's consistency.

A better structured QOC expressing the equivalent argumentation might be as follows:

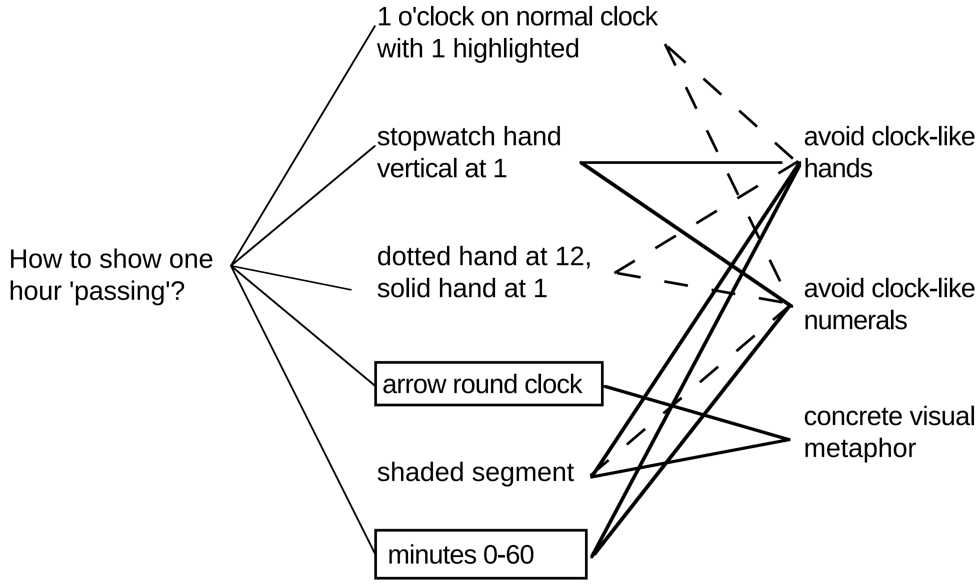


Figure 6.8: A well-structured QOC for the same argumentation as shown in Figure 6.7.

This representation factors out temporal information, introduces bridging Criteria to make explicit what it is about Options which was ‘misleading,’ and extracts features which were common to the Options in Figure 6.7. This is an example where selecting more than one Option seems to be the most elegant representational strategy.

Two strands of work offer possible ways to capture evolving designs and their rationales. One approach to the problem is described in the work of Goldstein and Bobrow (1980) on the Personal Information Environment (PIE). PIE was an environment in which descriptions of Smalltalk program structures could be developed and manipulated. It provided a mechanism for exploring alternative ways to implement operations via its notion of *contexts* (in which each implementational alternative was assigned its own context). Furthermore, each context could be structured into a sequence of *layers*, a mechanism for recording the evolution of descriptions in a context. Goldstein and Bobrow also discuss the use of layers to support the coordination and merging of parts of a design. The *rationale* for changes is attached to each new layer, thus easing its retrieval when integrating different designs.

Secondly, Reucker and Seering (1991) propose that the content of goals and alternatives can be formalised to enable DR (in a version of SIBYL) to be queried in ways which are not possible when knowledge is represented purely as entity and relationship types. By constraining the domain to mechanical engineering, and formalising the content of nodes to a greater extent, they show how states within the artifact can be represented within a ‘transition space’, adding greater power to SIBYL’s precedence management facility. The relevance to the present discussion is that transition space traces allow the transformation of design alternatives over time to be represented explicitly. Whilst this adds another expressive dimension to DR systems, it remains to be seen if designers will regard the added benefits as sufficient to merit the extra encoding effort.

6.3.1.3 Option dependencies between Issues

In software design on any significant scale, it becomes nearly impossible for designers to keep track of dependencies. One of the services which a DR tool should support is monitoring contingencies of this sort. In terms of QOC, dependencies manifest as constraints on the selection of Options in different parts of the space about a design; reproduced below are several incidents in which dependencies were encountered in this way.

(i) Pair 2 (ATM): The designers realise that an earlier decision (SELECT ALL [SERVICES] AT ONCE) which specified how to select multiple ATM services, now constrains decisions made subsequently about the order of interaction events, and have to recheck that the three are compatible.

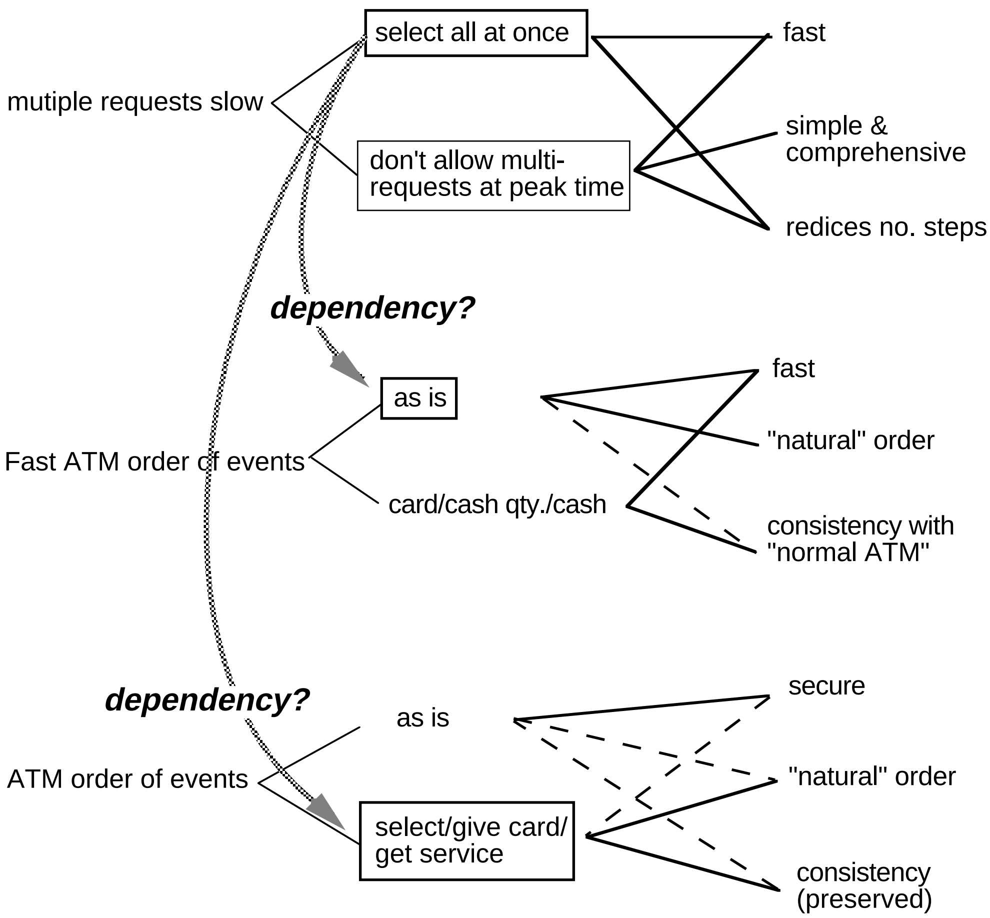


Figure 6.9: Realising dependencies between an early decision and two later ones (the arrows have been added).

(ii) Pair 12 (ATM): In this example, an explanatory note and '?' are added to highlight the contingency between an Assessment, and implementational decisions which have still to be made:

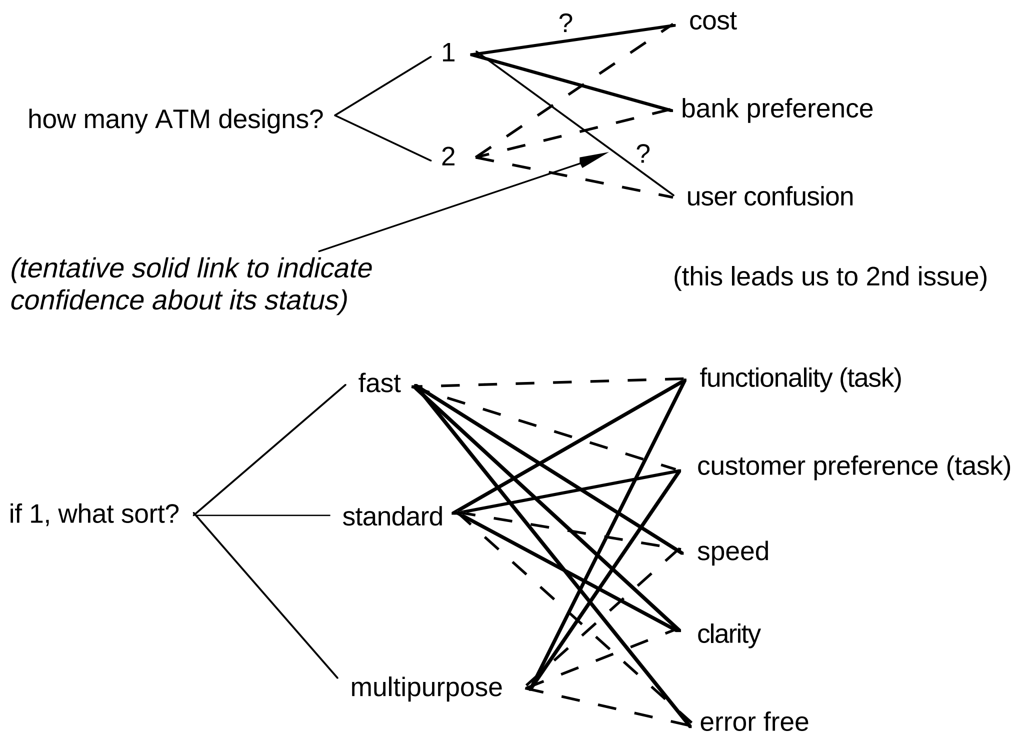


Figure 6.10: Annotating QOC to indicate dependency between two Issues: the Assessments which depend on subsequent decisions, are queried ("?"), and an explanatory note is added.

J	. . . so we're heading towards 1, because USER CONFUSION would be better for 1 (links in)... well now – that depends on this doesn't it (what they decide in the consequent Issue), but in terms of assuming that the Issue is just 1 machine or 2 machines?
G	it's hard to come to an answer here
J	but 2 could well confuse them, whereas 1... (links in with tentative solid line, and adds '?' over it). Because this one then (adds note under first Issue) *(this leads us to 2nd issue).* We don't know if this is going to confuse users, 'cos we don't know what sort we've got

The requirement to make dependencies explicit is satisfied to varying degrees by different DR notations. It has been suggested that gIBIS could represent dependencies between Positions by introducing intermediary Constraint nodes which bridge the Positions via *demands* and *stipulates* links (Conklin, 1989). SIBYL has the most powerful mechanisms for managing dependencies, this being a service for which knowledge-based tools are well suited. Lee (1990) describes how DRL would, for example, manage dependencies between choosing a programming language, and choosing from several different hardware platforms which support different languages.

Although MacLean et al (1989) make reference to internal and external consistency links, and briefly touch on the notion of 'imported' constraints from other decisions (MacLean et al, 1991), the concepts are as yet underspecified. For instance, it is not clear whether new link types need to be introduced, whether compatibility with other decisions becomes a new Criterion to be satisfied, or if an import is functionally equivalent to a Criterion. In whatever way this facility is actually implemented, the above examples add weight to the case that this is an important piece of vocabulary for a DR language.

6.3.2 Expressing tentative decisions

If a QOC tool is intended for use during design deliberation, it needs to be sensitive to the exploratory nature of design problem solving. Commitment is often delayed as alternative routes are partially developed to assess their potential as solutions. In one incident, a dashed box was drawn around two Options to indicate tentative selection of those Options.

Consequently, the binary *selected/rejected* distinction for Options needs to be extended, and dashed boxes as used by this pair would seem an effective visual device to this end. A similar requirement for the *supports/objects-to* dichotomy in Assessment links is noted below.

6.3.3 Criterion weighting

Many comments were made by subjects on the lack of a more sensitive scheme within QOC for evaluating Options. The notation as presented to them allowed for simple *supports* or *objects-to* relationships between Options and Criteria which proved too simplistic. Although closely related, conceptually speaking, requirements for weighting can be divided into the need to weight *Criteria*, and *Assessments*.

Designers pointed to the need to prioritise Criteria, and observed that simply counting *supports* links was clearly an inadequate way to make decisions. In terms of the overheads of introducing a weighting scheme, one pair of designers did not envisage problems, claiming that assigning ratings would not be difficult, and that this would encourage clearer, more objective thinking. This is an empirical issue.

6.3.4 Assessment-link weighting

Whilst Criterion weighting was important, the majority of incidents and comments on QOC's representation of the evaluation space related to the need to show relative weighting of Assessments. Designers would describe Assessments as being 'more positive' than others, and invented their own graphical conventions for encoding link strengths such as double-links or thicker links to represent gradations of the relation *supports*. MacLean et al (1991) do not consider numerical Assessment weighting a useful avenue to pursue (as Conklin and Yakemovic put it, "allowing [quantitative] measures into the DR places it on the slippery slope of decision support systems"). However, it is clear from this data that some way was needed to indicate at least *relative* (rather than absolute) Assessment weights.

Pair 6 made extensive use of ‘neutral’ Assessments, signified by *omitting* to link an Option to a Criterion. These subjects used ‘neutral’ links eleven times during the three exercises, in several different circumstances. Generally, it served as an Assessment for Options which were judged to fall ‘in between’ other Options which had positive and negative Assessments, that is, it was not judged to be sufficiently good or bad to merit a +/– link, as shown in the two examples in Figure 6.11.

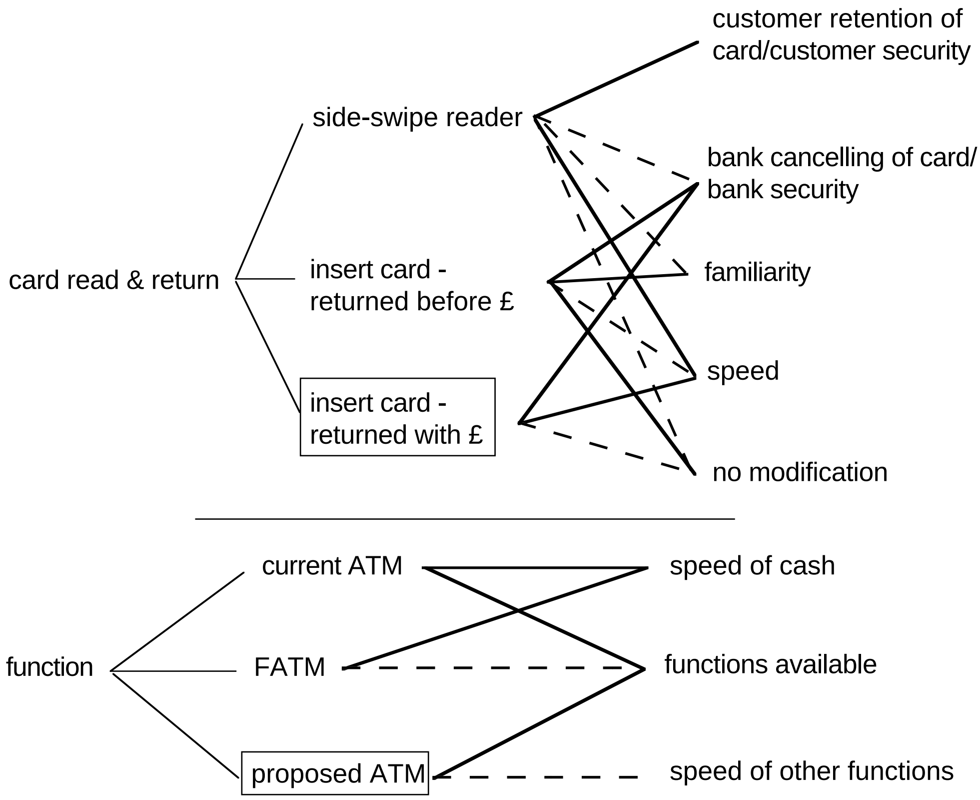


Figure 6.11: Two examples of the ambiguous use of ‘blank links’ to indicate neutral Assessments

In the lower example in Figure 6.11, since CURRENT ATM’s additional functions (i.e. services) are a menu level down, it is neutrally assessed by SPEED OF OTHER FUNCTIONS (i.e. it was judged to be better than PROPOSED ATM, but not to the extent that it merited a *supports* link). However, the FATM (Fast ATM) Assessment against that Criterion was also left blank, but in this case because the Criterion was simply irrelevant (the FATM only offers a cash service). This exemplifies the ambiguity of using absence of links to mean neutral. Elsewhere, neutral Assessments were used to mean still other things (*not yet decided*, and *not discussed*).

Another elaboration of Assessment links was used by Pair 12, who placed ‘?’s over links to represent undecided status as they worked (Figure 6.10). This occurred under two circumstances: (i) when an Assessment could not be made until further design details had been finalised, and (ii) when it was not clear if an Assessment was correct (because the designers did not feel qualified to make evaluative decisions about psychological Criteria such as display clarity or attentional requirements).

The above data demonstrate that the core QOC vocabulary needs to be extended to permit the representation of *degrees* of Assessment, and weighting of Criteria. The Assessment vocabulary should also ensure that *absence* of an Assessment link is unambiguous, to mean for instance, that the Criterion is irrelevant to that Option.[7] Depending on the representational medium, double/treble links, link thickness, numeric link labels, and colour are all candidates as visual correlates for intensity of Assessment.

6.3.5 Cognitive overhead of using weightings

The question now arises as to how disruptive the weighting process might be in different contexts. The limited evidence in this study demonstrates that designers often need to express relative Assessment and Criterion weightings when using QOC. This felt need would certainly play a part in motivating designers to use any weighting facility which a DR tool offered, but only up to a point; if the process is too cumbersome at the level of the user interface, or simply in terms of maintaining consistent use of weightings by different designers, then potentially serious authoring and interpretational difficulties arise. Let us consider Assessment and Criterion weighting in turn.

Deciding the extent to which alternatives meet goals is an activity in which designers have to engage as a matter of course, and the QOC version of this task – Assessment weighting – is simply one means of reifying those judgements. Assuming the user interface to a tool is well designed, this and other studies suggest that this task should not be intrusive.

If one wishes to automatically propagate weightings, the situation becomes more complex. Lee’s SIBYL system has the most developed representation of ‘plausibility’ weightings of relations between Alternatives and Goals (equivalent to QOC’s Assessments). DRL can use numerical weightings (e.g. 1-5) or nominal categories (low-medium-high), but as Lee points out, the algorithm for propagating and merging evaluation such measures must also have validity. There is no evidence as to whether designers use, or find it easy to use this facility in SIBYL. It is certainly the case however that use of SIBYL in asynchronous as opposed to real time would give users more time to explore the implications of different weightings.

The QOC principle of designing to a set of prespecified Criteria suggests that general Criteria could be assigned weights at the start of the project, project phase, or meeting, such that dynamic assignment of weightings to key Criteria *during* meetings would be less of a problem—they would already be decided. This is not to preclude the possibility of modifying weightings as new Criteria or unforeseen factors come to light.

---

7 As §10.3 explains, however, this should not happen often – well structured DR generally asks Questions which spawn Options at a common level of detail and of a sufficiently similar nature that each Criterion can meaningfully assess each Option.

Perhaps the key question for both of these uses of weighting, is people's ability to consistently use rating schemes. Research on the use of scaling techniques (e.g. Ghiselli et al, 1981) is potentially relevant here. Much effort has been spent on studying how people assign ratings, and techniques have been developed for coping with the inherent variability in people's judgements. One difficulty which may merit further investigation is that in the behavioural sciences, ratings for an item are almost always gathered from a large subject population, or from one subject who assigns repeated ratings to the same item over an extended period; this allows the inevitable variance to be normalised. In the context of DR, an Assessment weighting would, for instance, be assigned by only one person on one occasion, creating potential error problems for computational analysis. One example is calculating the most favoured Option on the basis of the Arguments for and against each of its Assessments, as in SIBYL's plausibility management scheme (Lee, 1990). Devising a usable weighting scheme which allows reliable comparisons using absolute as opposed to relative values is an open problem.

6.4 Conclusions from Study 3

This study was the most comprehensive of the six conducted, in terms of the number of designers studied, the depth of analysis of the QOCs generated, and the classification of usage patterns. The laboratory study of design has proven itself a useful paradigm for detailed video-based observation of behaviour in a poorly understood domain.

Study 3 has documented what might be characterised as the 'nuts and bolts' of using a semi-formal, argumentation-based notation—the core cognitive tasks and representational mechanics involved in transforming into QOC ideas which arise in design deliberation, and the sorts of difficulties which are typically encountered. The data here are supported in the observations of QOC-use reported in the remaining studies, and are related to the few studies of DR use conducted elsewhere (reviewed in §10.1).

Chapter 7

Case Study 1: Using QOC in Two Different Modes of Design

7.1 Introduction and method

7.1.1 Introduction to Case Studies

The following chapters present three case studies of QOC in use. Whilst Studies 1-3 simulated realistic tasks to some extent, the case studies represent a move towards taking QOC ‘out of the lab’ in order to test some of its assumptions against designers working on real problems. The results of each study include a brief description of the design project, the methodology used in the study (which varied in each case), discussion of the results, and some conclusions.

7.1.2 Case Study 1: The designer, and design project

In the first case study, the designer (a second year doctoral research student) working in a music-technology research group, agreed to be trained in QOC, and to be video-recorded over several sessions whilst using it in his everyday work. The designer had seven years’ programming experience in total, with about 1½ years’ experience in Smalltalk-80, the language used in this study.

The aim of the project is to develop a music composition system which can control any kind of synthesis hardware such as the widespread MIDI-controlled[1] commercial synthesisers or custom DSP-based systems. The system being built should enable a score composed on one device, to be realisable on any other device with equivalent functionality. At present this is not possible in electro-acoustic music, as scores are specific to the particular equipment on which they are written.

7.1.3 Methodology

The designer was introduced to the concept of DR, and to QOC via a shortened version of the tutorial used in Study 3 [§5.1.5]. He tackled training exercise 1 [Appendix 5] involving analysis of extracts of design discussion which were discussed afterwards (40

1 MIDI: Musical Instrument Digital Interface. DSP: digital signal processing.

minutes), and training exercise 3 [Appendix 6], analysis of the scripted-video, which working alone took about one hour. An informal assessment is that this designer was one of the fastest in learning QOC, producing relatively well-structured representations (e.g. positive Criteria, and well formulated Questions addressing one issue at a time), and showing a keen grasp of ways in which to use and construct QOC representations.[2]

Three design sessions were recorded (each lasting for about 1 ¼ hours) over a period of 3 ½ months, the first Session two weeks after training, and then at approximately equal intervals. The design work recorded was not selected according to any strict criteria, except that there should if possible be a variety of different kinds of design work (see next section). Although the designer was encouraged to try and use QOC during his work in between the recorded sessions, he reported that he had not found it necessary or useful. This may have well have been due to the nature of the design in which he was engaged which was sampled twice (Sessions 2 and 3), and which as described later caused serious difficulties for QOC [§7.2.2].

A video-recording configuration similar to Study 3 was used [§5.1.6.2] except that the experimenter was present in the room controlling the camera and interacting with the designer. The designer described his work and use of QOC to the experimenter, who simply observed much of the time, but pursued potentially interesting avenues as they arose; thus, the protocol became a discussion at points.

The design problems for Session 1-3 are described prior to each of the two analyses (Session 1 versus Sessions 2 and 3), since the problems are closely tied to the representations and modes of working which determined the role of QOC.

7.2 Results and discussion

As the designer's comments constitute an essential part of the data, extensive use is made of transcript extracts in order to illustrate points. The results from this case study are organised into two main sections. The first reports the first session run with this designer, during which QOC was used to represent the solution and rationale to the problem in hand. The second section deals with Sessions 2 and 3, in which the designer encountered serious difficulty in using QOC. The reasons for the apparent incompatibility between QOC authoring and his mode of working are considered.

2 One might speculate that object-oriented design experience affords an advantage over other designers in learning to use DR. The emphasis in DR, particularly design space analysis, on clean conceptual structure and meaningful naming of entities, requires very similar skills. It is true that all software design involves this to some extent, but the object-oriented paradigm makes it more explicit than other approaches.

7.2.1 Using QOC to make 'strategic' design decisions (Session 1)

In Session 1, the designer was making a decision about the approach he would take to allocating incoming system 'events' to the limited number of instruments available. He referred to this as "strategy decision-making" between several alternatives which traded off against a number of competing factors. In contrast, Sessions 2 and 3 represented the implementation of the strategy adopted in Session 1.

As in previous studies, examples of *renaming* and *restructuring* were evident, as was the need to support *weighting of Assessments* and *Criteria*. These activities are described briefly in the following sections.

7.2.1.1 Naming/renaming nodes

As with subjects in previous studies, finding good names for nodes represented a not insignificant overhead for the designer. Names were modified on several occasions either because they no longer represented a valid idea, or because other similar concepts had arisen subsequently, and the names needed to be differentiated.

7.2.1.2 Structuring/restructuring

Copies of the QOC structures were given to the designer after they had been redrawn using graphics software (part of which is shown in Figure 7.1). This made them much clearer, omitting deletions, but preserving useful annotations to the structure.[3] The designer reported that this redrawn QOC was extremely useful.

The following sections describe how the the QOC was restructured during the session.

---

[3] The changes made during the DR were omitted in the redrawn versions, to convey the final DR most clearly. Changes of specific interest are reproduced in Figures 7.2 and 7.4.

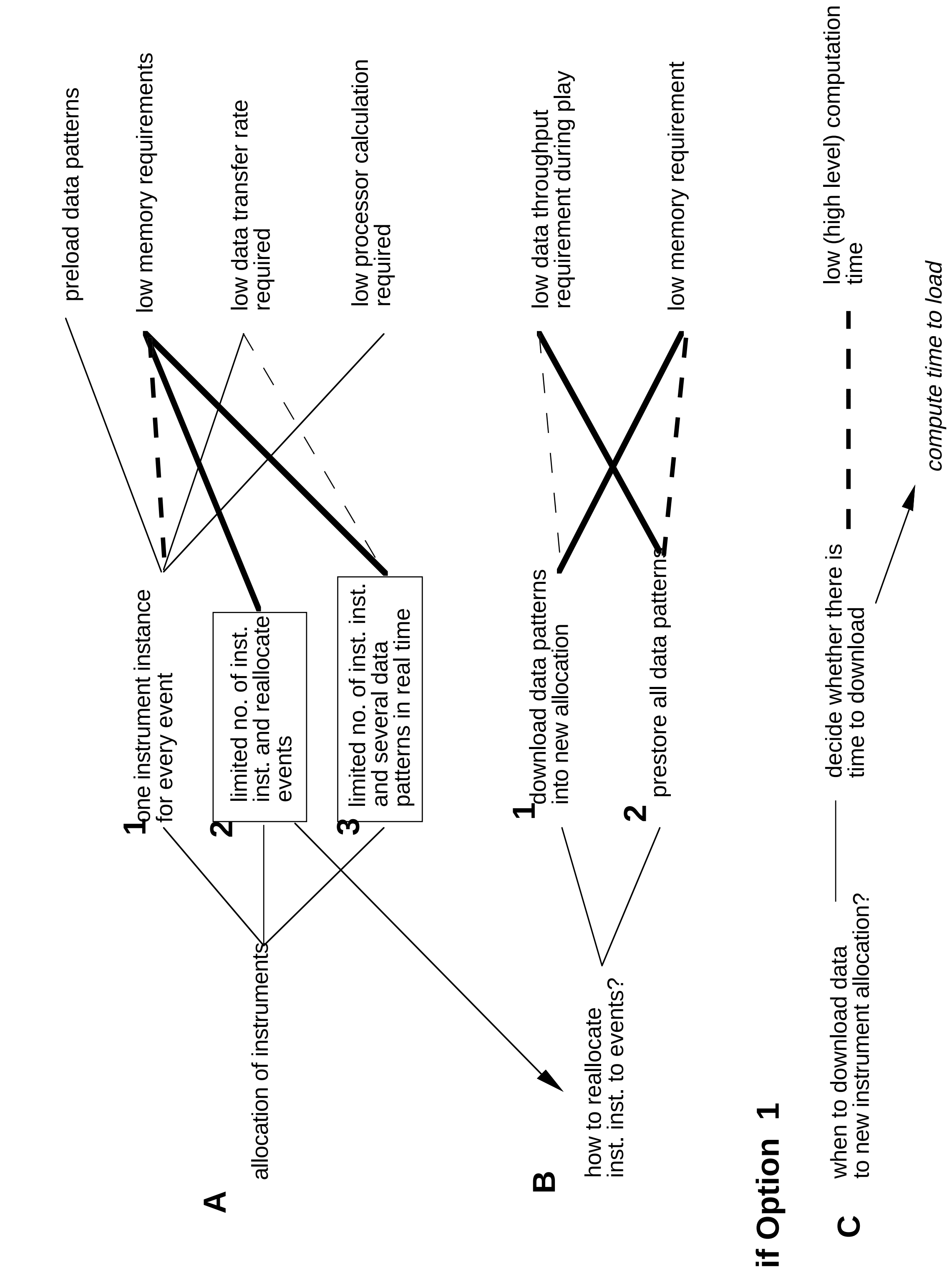


Figure 7.1: Part of the QOC analysis produced by the designer in Session 1, redrawn for clarity.

7.2.1.2.1 Options

The QOC structures in this study centred around the decomposition of Options. Initially, this took the conventional form of a consequent Question (Figure 7.2, an extract from Figure 7.1). This proved necessary when he found that he could not assess an Option without further exploring the details of its design.

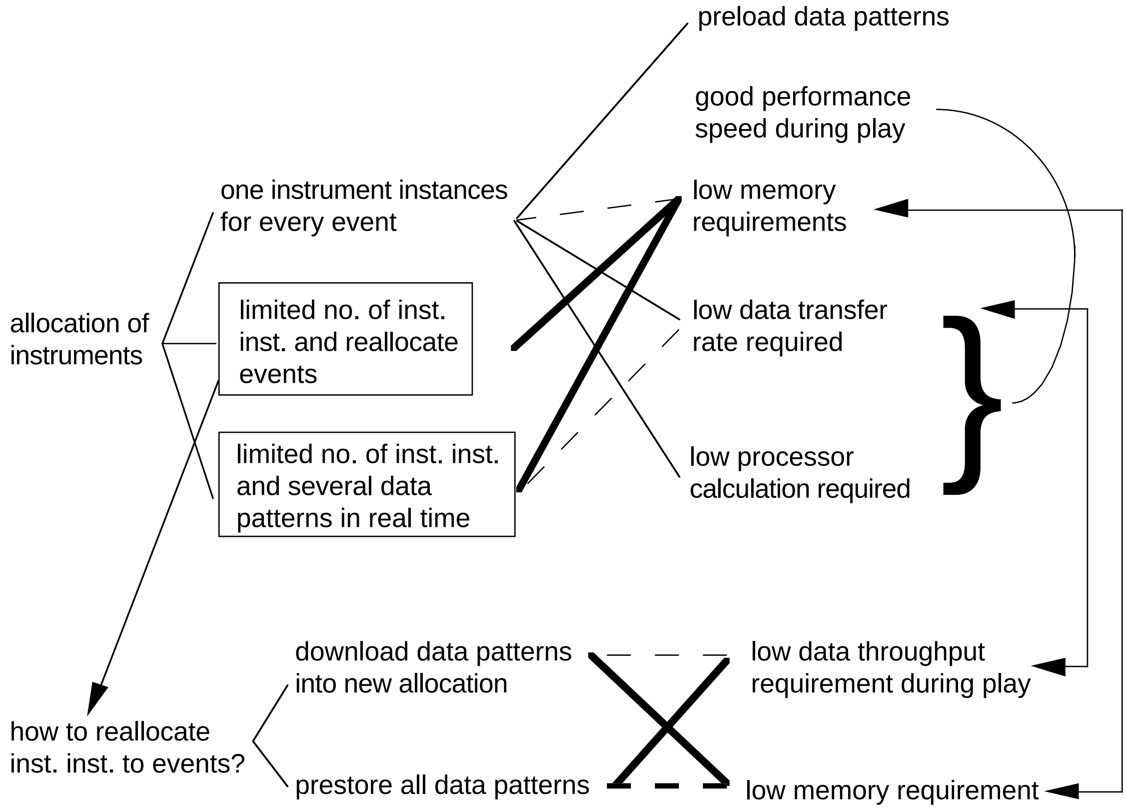


Figure 7.2: Part of the QOC (from Figure 7.1) showing Option decomposition through a consequent Question, and the inheritance of Criteria from the parent Question (the designer added all of the annotations shown).

In Figure 7.2, the middle Option *limited no. of inst. inst. + reallocate events* is not assessed by the Criterion *low data transfer rate required*, as the designer needed to know *how* events should be allocated. This was considered in the consequent Question, where it was found that *low data throughput reqt. during play* supported one Option but objected to the other. Note the decomposition of the Criterion *good performance speed during play* into the more precise bridging Criteria *low data transfer rate required* and *low processor calculation required.* The above example also illustrates how lower level Options inherit the Assessments of their parent Options (as shown, the designer suddenly realised he was reusing the Criteria *low data transfer rate* and *low memory requirements* in the consequent Question, and marked them in).

In order to produce a single representation which summarised the above Option decomposition, the designer decided to show the Option refinement explicitly. Figure 7.3 shows the new structure, with the Option hierarchy in the first two Questions more clearly. (The Options have been numbered for ease of reference). It can be seen that the new structure was still complex, even after having been laid out to make relationships as clear as possible.

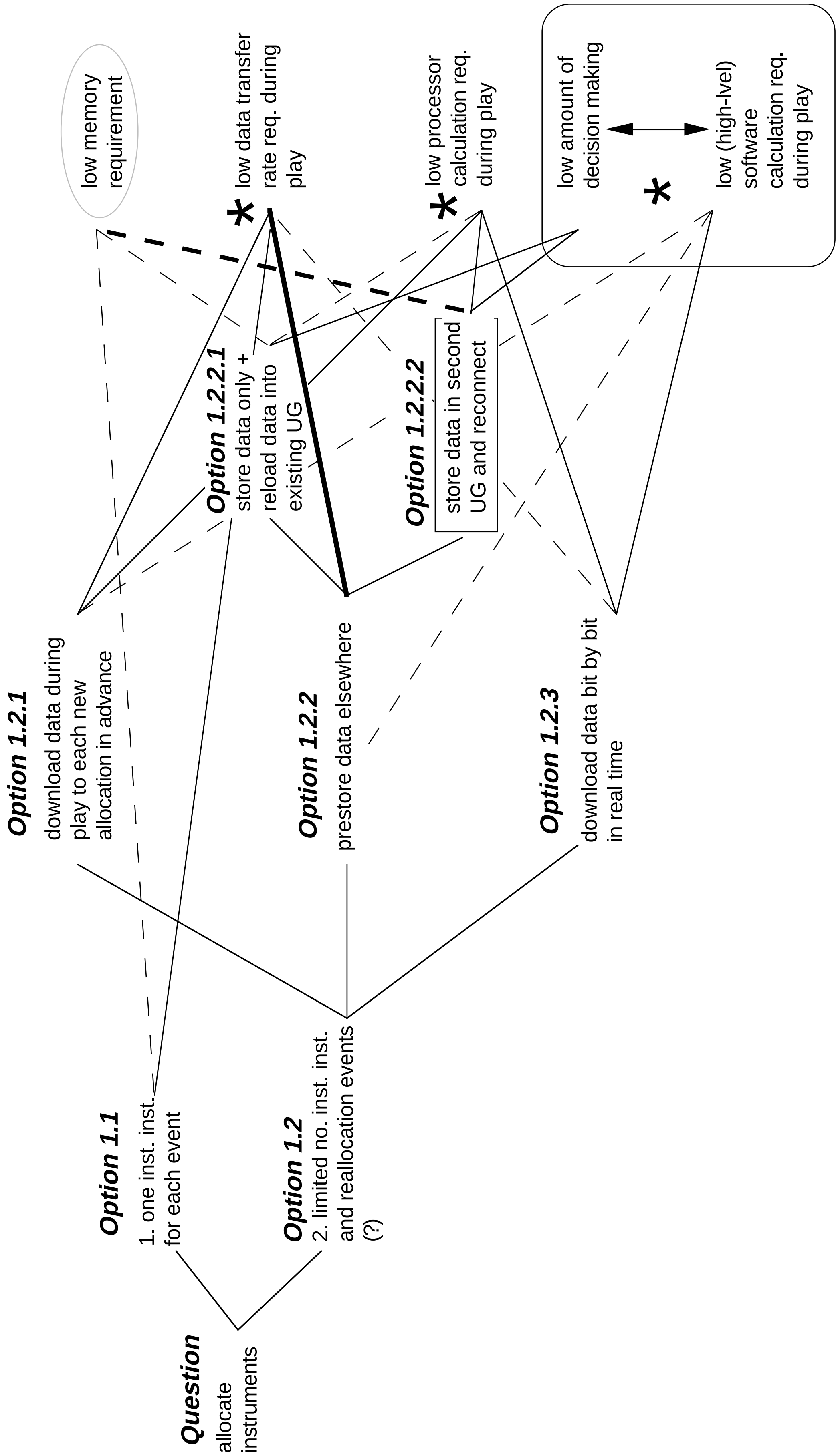


Figure 7.3: Making the Option hierarchy explicit.

In the process of redrawing, additional insights were gained:

- another level of Option decomposition was added, by *refining prestore data elsewhere* (Option 1.2.2 → 1.2.2.1 and 1.2.2.2)
- it was realised that the third Option to the first Question, was really another refinement of the second Option (i.e. Option 1.3 became 1.2.3). This Option was therefore moved down a level in the Option hierarchy.

If 'structural transformations' or 'manoeuvres' of this sort recur frequently within QOC (and other DR notations), tools should automate them as standard operations. In the above situation, therefore, the designer might declare that Option 1.3 should be Option 1.2.3, and the tool would redraw the structure. Since Assessment strengths are relative to others at the same level [§6.3.4], strengths may change when the level of an Option changes; the tool could therefore prompt the designer to confirm the strengths of the Option's Assessments in relation to its new sibling Options.

7.2.1.2.2 Criteria

An analogous situation related to Criteria. When Criteria are reorganised, the Assessment links need to be redrawn, which cannot be done easily manually. Often, relationships between Criteria are only understood *as* the Criteria are used during design – the optimal structure reveals itself as part of the process (Fischer, 1988). Consequently, just as with Options, tools should support common restructuring activities with Criteria.

Criteria are always restructured following the identification of new relationships between them (e.g *is-subsumed-by,* or *depends-on*). This happened twice in the relatively small DR generated in this study, suggesting that it may be a common phenomenon. In one instance, a *parent-child* relationship was recognised between the general Criterion *good performance speed during play* and the bridging Criteria *low data transfer rate required* and *low processor calculation required.* This general Criterion was then deleted, along with its Assessment links.

In the second instance, identification of a *parent-child* relationship between Criteria led to them being merged. Figure 7.4 shows two Options and three Criteria. The designer decided that (the top Criterion) *low processor calculation reqt. during play* was a child of the more general Criterion *low amount of decision making*, and grouped them together. However, he then realised that in fact they were independent, but recognised another relation instead—that *low amount of decision making* served (the bottom Criterion) *low (high-level) software calculation reqt. during play*. He 'unboxed' the first pair, and boxed the second pair (see Figure 7.4). This was an interesting conclusion to have reached, given that the Assessments were not identical for each of these Criteria, as one might have expected if one was a direct consequence of the other. Nonetheless, the designer was clearly following this logic, as he proceeded to delete duplicate Assessment links to the constituent Criteria in the new composite.

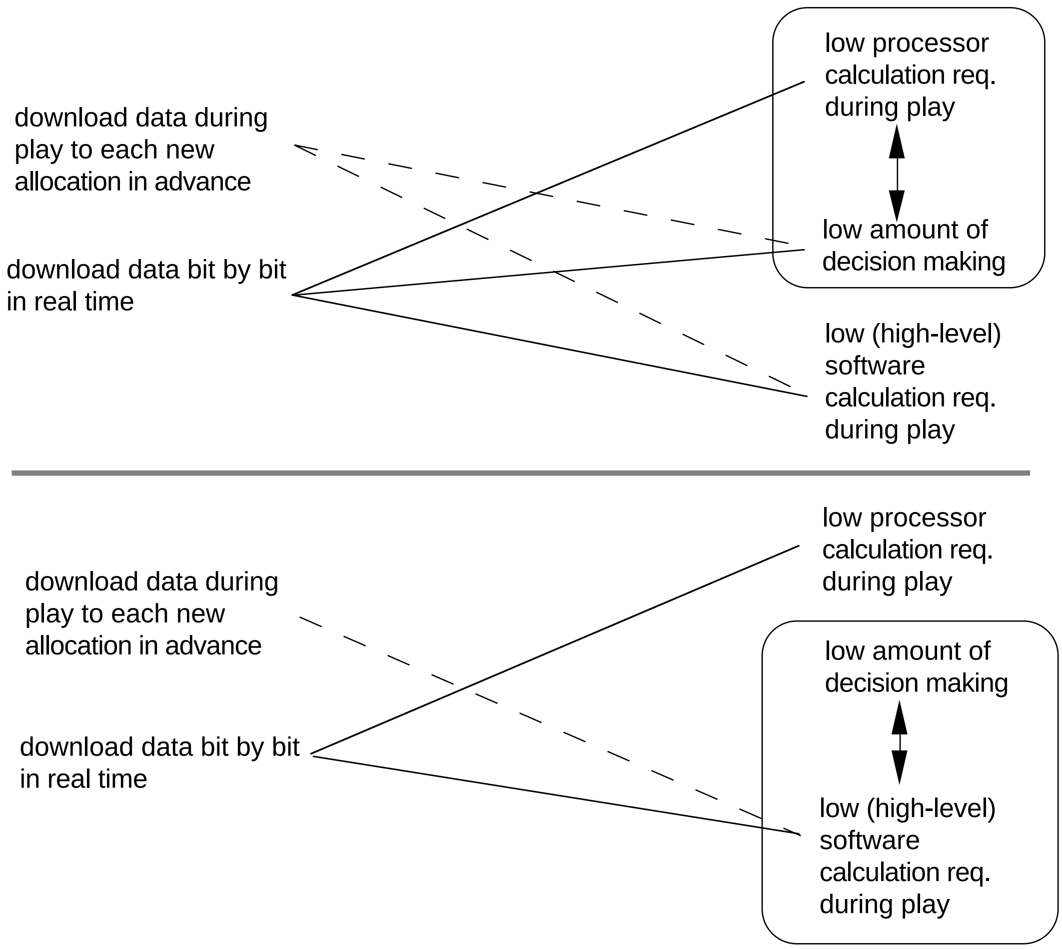


Figure 7.4: Options and Criteria from part of a QOC before and after regrouping Criteria. Note that duplicate Assessment links to the constituents of the new 'composite Criterion' have been deleted.

As in the Options example above, a tool could clearly assist in these representing these reconceptualisations, in this case by enabling Criteria to be 'chunked' and 'unchunked' with minimum effort. One might also envisage automatically updating the Criterion tree following such changes: in the above case, the Criterion *low amount of decision making* would be moved under or linked to its new parent Criterion.

7.2.1.3 The need to weight Criteria

The importance of being able to weight Criteria was highlighted in a new way, when in Session 2, the designer reconsidered the DR constructed 1½ months earlier in Session 1.[4]

The Criterion *low memory* had been assigned a low weighting because it had been assumed that memory could be expanded cheaply, and as such represented a small

[4] This example demonstrates the advantages of studying DR use over longer periods than Studies 1-3 permitted.

obstacle to memory-intensive Options (Figure 7.1). In Session 2, however, he noted that the weighting was inappropriate:

> (studying the DR) so *low memory requirement* – that turns out to be more important to have low memory than I thought at first. Apparently this kind of quick memory is rather expensive, so it's not just a load of sims as I'd thought at £30 a time!
>
> So it would be nice to have some way of grading the importance of these (indicates Criteria). So that negative link (to Option 1) is more important than I first thought – but that's good – reinforces why it should be rejected. So it would be useful to grade importance of the links as well – that negative link is a lot more important than that (indicates a positive Assessment from the rejected Option to another Criterion)

7.2.1.4 Benefits of representing DR

DR will not be used by designers if they do not perceive that they are benefitting from it in terms of the own work. It will be insufficient to reassure designers that it is worth authoring DR to make their work easier to understand by others.

To what extent did use of QOC assist the designer? Although Session 1 lasted less than 1½ hours, there were a number of incidents in which QOC appeared to play a constructive role in clarifying ideas; interestingly, most of these related to defining Criteria.

Several times, it was not clear what the difference between two Options was in terms of the Criteria, even when the designer knew that he had already rejected one Option in his own mind, that is, he was trying to rationalise an Option he already favoured. The process of working it out sometimes lead to developments, for example in one case to restructuring the way in which the Options had been decomposed (Figure 7.3), and in another to a new Criterion. On another occasion, the designer first confused, and then clarified which Criteria were relevant (illustrated in Figure 7.4).

In the extract below, Criteria are made explicit by having to think more precisely about what an existing Criterion means:

> It's got *low memory requirement*, but there must be something wrong with it or I'd be keen on it. *Performance speed* is a bit wishy-washy – what do I mean by that? *Low data transfer requirement*... also *low processor... calculation requirement*

On completing the QOC, the designer evaluated the final Options in order to made his decision, and having all of the Options and Criteria laid out clearly assisted in this task. In discussion afterwards, the designer summarised his experience with QOC (the first application to his own work) as follows:

> I may have made that decision anyway, but I certainly wouldn't have been so aware of the reasons why I took it, and I wouldn't have been able to answer people's questions about the other alternatives. I still find the notation difficult for some way of weighting the Criteria – for both degree of connectivity and importance. No but it's.. you always have to re-interpret what you've written and then make more decisions about those things rather than just leading automatically to a solution.

> *E: was all that stuff that you already knew and were just putting down in DR format, or were you covering new ground?*
>
> All the Options there were things I'd thought about – I had had them in mind. What has come out is the sort of .. the different subCriterion [he may have meant sub*Option*] ways of doing things. I had thought of that Option there but hadn't realised there were these [three] separate ways of doing it – I certainly hadn't examined all the ramifications of doing it in these different ways. (referring to the Option decomposition – Figure 7.3)
>
> If I hadn't already known the problems with these things (Options) I don't think they would have just come out of nowhere – but it certainly enabled me to organise it better. I'd like to have that diagram now, I'd find it useful for later thinking. Of course if I take this Option and find out there are further problems, I'll want to go back to the next best Option (indicates returning to the DR)

In Session 2, the designer was reviewing the Session 1 DR, and commented on its value:

> That was very useful the more I think about it – I didn't really know what to do, and everyone's agreed with the decision since, so it was obviously a sound decision to have come to, and I'm confident that it was the best way of doing it.

To summarise, in this problem, QOC was not used to generate initial ideas, largely because a lot of thinking had been invested in the problem beforehand—the main task was rationalisation and decision making. Nonetheless, there were incidents where advances in thinking were reported by the designer as he developed the QOC analysis. Patterns of QOC authoring observed in the previous studies were replicated, providing evidence that they apply across domains, and at a different level of software design from user interface issues. The designer reported that QOC had enabled him to organise his ideas into a clear rationale which he could then present to others.

7.2.2 A QOC-incompatible mode of design? (Sessions 2 and 3)

In Sessions 2 and 3, serious difficulties were encountered in using QOC, to the extent that no explicit DR was created. The root cause appeared to be that the problems the designer was tackling demanded a mode of working in which it was extremely difficult to decompose the problem into subissues (i.e. Questions) or identify discrete alternatives (Options).

7.2.2.1 The mode of working demanded by the design problems

In Session 2, the designer was constructing the data structure such that incoming commands with high level sonic parameters (e.g. pitch, volume) could be translated into inputs for low level components like oscillators and reverb units. As an aid to understanding the way in which the designer worked, Figure 7.5 reproduces the main working representation of the data hierarchy developed in Session 2 (the indented list). The arrows indicate message passing between data types.

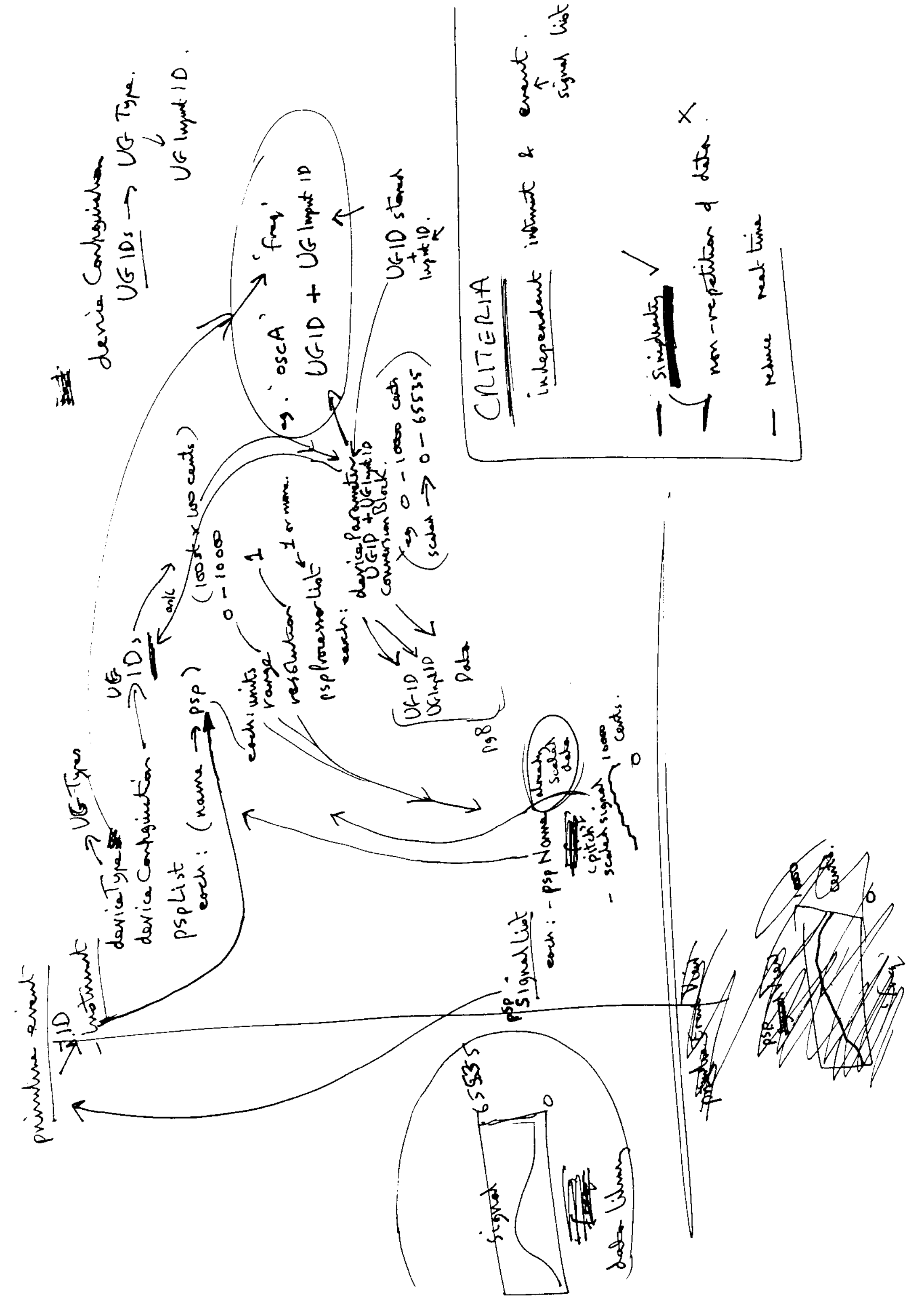


Figure 7.5: The data structure plan constructed during Session 2.

In Session 3, the problem was similar, in this case to design a data structure which enabled a hierarchical description of any hardware synthesis structure to be constructed in Smalltalk. The main working representation was a matrix of cells containing data types, in which each column was a decomposition of the preceding one.

In Session 2, the designer described the method of deriving the hierarchy as follows:

> What I'm doing is a sort of consistency check – thinking through the implications of what I'm doing – this draft suggestion here. And I'll incrementally alter things – I mean I've already done that many times to get to this stage...

> (points out that he's refining an earlier sketch from his notes) 'Gradual refinement' is the phrase. I don't know the Options until I test the previous Option (laughs)

Descriptions of the process in Session 3 are indistinguishable:

> I'm now doing Smalltalk style 'pseudocode' – programming level thinking...

> ...so what I'm doing is imagining a suggested setup – a structure – and then I'm going to go back (circular gestures to indicate iteration round) to make sure it'll do what I want it to do, in terms of holding the right structure in the right place.

> ...I may then slightly change it, and if I get into too much trouble, I'll have to go back and start again, but hopefully it'll be just small details.

> I'm postulating a structure, going round and round testing it, and drawing a few example diagrams of applications of that structure to a real situation – getting it into some concrete familiar objects ... checking that the abstract structure fits that, and changing it if it doesn't.

To summarise, the design problem solving in Sessions 2 and 3 was characterised by:

- ❑ opportunistically driven generate-evaluate cycles to hone the form of the message passing hierarchy;
- ❑ use of complex, concrete examples as test cases for the abstract structure;
- ❑ management of numerous constraints within the message passing hierarchy;
- ❑ application of much implicit Smalltalk programming knowledge.

These features corroborate existing research into object-oriented programming cognition (e.g. Visser, 1990; Booch, 1991). Given that this is the way in which the designer was working, attention now focusses on its apparent incompatibility with an argumentative mode of design, or retrospective rationalisation.

7.2.2.2 Problems using QOC in evolutionary design

What were the implications for the use of Questions, Options and Criteria in Sessions 2 and 3? The designer's experiences with the three main QOC constructs are illustrated below with extracts from the verbal protocol which are then summarised as key points.

7.2.2.2.1 Questions

> What are the decisions that I've made? (tries to formulate Question) *Do you have* ... it's difficult to put it in terms of that kind of Question... *is a device configuration a separate class or just one type of category?* Now I don't know how I answer that – it just fits in that it's... there's no real doubt about that, it just fits in consistently.

> *E: So you can make up a Question if you have to?*

> Yes in some cases. Let's see if I can make up any more then ... this idea of having a category of submodules which it's allowed to have – that just arose as a solution to a problem – and what was the problem? The problem was that you have a structure where each object can't just have any old object that's available as its child, only selected ones. And there may be many children ... [goes on to describe relationships between types]

- ❑ There are two ways in which Questions can be used: either by posing an extremely general Question such as *what is the best data structure for a primitive event?* or through a long series of Questions each of which records the particular problem on the each iteration.
- ❑ A general Question offers no analytical power to the designer, and no insight to someone else trying to understand the design.
- ❑ The implicit nature of the designer's expertise would make explicit recording of the hundreds of context specific Questions unrealistic – the consequences would be enormous DRs, with a proportionate increase in authoring overheads.
- ❑ QOC Questions are meant to pick out generic or important dimensions of the design—it is therefore difficult to see their useful application in this context.

7.2.2.2.2 Options

> [Sess.2] but as for articulating possibilities – they only arise consecutively. I couldn't have initially said we've got two ways of doing it: like I've done it for those messages, or 'that' – now let's think which is the better one ... the only way you arrive at the second one is by having the first one there and thinking now still what's wrong with that? You go more in a linear way than a bifurcated way being implied by your DR scheme ... as it's a linear, iterative, refining kind of design, the Options are less useful.

> ... bits are useful – Criteria. Options are hard to parcel up – often very similar but for one detail. It's almost a problem of notation: how to record each stage, as it's an iterative process.

> [Sess.3] Am I making decisions? I must be... about organisation. And I'm trying to get consistency, so consistency you could say was a Criterion ... but I don't see that there's a single parameter where I could say 'well we could have this Option or that Option, that gives me consistency, this doesn't.'
>
> It's a lot of spaghetti – it's a big knot I think – you have to get one bit loose, and another bit, and make sure that hasn't tightened up there... There's no set of Options – it's just one big mass you have to sort out into categories.

- ❑ Discrete Options were impossible to identify because the final design was effectively the evolution of one Option over time; the difference between each Option was only one, or a few fine details; there were effectively tens of versions marking the path to the final design.

7.2.2.2.3 Criteria

Criteria were the only elements of QOC which were found to have 'face validity' in Sessions 2 and 3, that is, only Criteria seemed to be playing an active role in problem solving.

In the extracts below, the designer tries to identify Criteria which he has been using up to that point, and is able to identify the main trade-off between *simplicity* and *non-repetition of data*:

> Now do I have any Criteria? (sighs) I suppose I've made some decisions. (creates area headed 'Criteria' in bottom corner) I wanted to be able to have independent – I haven't really thought about it overtly, so let's see if it's useful.
>
> I want (writes) *independent instrument and event.* The event's got a *signal list,* and I don't want them to be tied together. I suppose I'm trying to have the simplest data structure possible – *simplicity.* I don't want repetition – *non-repetition of data* – I'm trying to express all these positively. On the other hand there is a slight conflict between *non-repetition* and *simplicity* of data; to have non-repetition you have to have lots of references to things, which can make it a lot less simple.
>
> *E: Do you have a general policy on that for the whole design, or does it depend on each situation?*
>
> In a way I'm still learning about it. As Smalltalk has pointer references anyway, it's actually quite efficient. I can put a new object in, and unless I do a deep copy of the object, it will just be a pointer anyway, so it won't be that... so I think really *simplicity* I'm coming down to is the key element, and I don't care about data apparently being repeated, because it won't actually place much overhead on the system. So *simplicity's* more important than *non-repetition.* (heavy underline of *simplicity* plus tick; crosses out *non-repetition*).

This incident may be an example of the benefit of explicitly considering Criteria; having enlarged on each of the Criteria, the designer concludes that in fact *non-repetition* can be easily satisfied within the Smalltalk environment. Although there is little doubt that he knew of this Smalltalk facility, and that *non-repetition* is a general principle in object-oriented programming, it is not clear if up to that point, the connection had been made in his mind, such that he no longer worried about data repetition as a problem. Once the Criteria were written down, he referred back to them on several occasions when he recognised that he was using them.

However, the designer made the following comments about explicit recording of Criteria:

> I don't know *how* I'm making half of these decisions. I think it's a whole block of expert knowledge – well experience – that I've built up of object-oriented programming; having seen examples, it's very difficult to articulate every reason for everything.
>
> [turns to Criteria noted during the session] Here we've come up with some Criteria – again useful to have those down, but a lot more difficult to go back over this and explain to people why I've done it this way. All I can say is 'well this one works at this stage.' Difficult to go back to another branch and say well this didn't work because ... everything would work (ie. previous versions) but this simplicity idea, is a difficult one to then give alternatives to – it's a subtle one

Thus, although it was useful to have Criteria recorded, the difficulty of using any additional reasoning structure (Questions and Options) meant that Criteria could only be referred to in general terms, applying to the whole data structure which constituted the 'Option.' His comment that "everything would work" reflects the fine-tuning process in developing successive versions of the structure.

In session 3, the theme continued:

> I'm not seeing DR ... as needed at the moment

> *E: why is that?*
>
> Am I making decisions? – I must be... about organisation. And I'm trying to get *consistency*, so consistency you could say was a Criterion ... but I don't see that there's a single parameter where I could say well we could have this Option or that Option; that gives me consistency, this doesn't...
>
> *E: Questions are meant to pick out the different aspects of a problem which make it complex. Is it possible to articulate different aspects of the problem?*
>
> I'm struggling to articulate different aspects of it ... there are two Criteria at the moment – that it's *consistent*, and *flexible* enough to cope with any situation, so I'm deliberately choosing quite a complex situation to model, 'cos I'm pretty sure the simpler ones will fit.

Later, the designer commented that it was impossible to explain why a structure was 'good' at the level of individual decisions, because there were so many dependencies. In the final analysis, the only Criterion was 'did it work?':

> ... now there's a very complex relationship as to why that's better – I sort of just hit on it 'cos it seemed to fit in – it just sort of happened. I mean I know how I got it – by working round the problem, drawing examples, and you just get a feel... you abstract from the concrete examples into the structure that will ... sort of inductive, whereas I think DR is deductive.
>
> ... this sort of thing, you're looking at a set of outcomes that you want it to do at the end, and you've got a lot of different ones, so you work almost backwards to possible ... you're still evaluating them – you've still got Criteria, but all the Criteria is, is 'does this structure I've just created enable me to go back and ...' – you make a rough guess at the structure from the examples, and go back and just test it rather than having Criteria; it's sort of simpler than DR perhaps.
>
> ... the only Criterion is, does it enable, or not? There's not a set of things it could fulfill – it either does or it doesn't. If it doesn't enable you to create a detailed concrete structure, then it's no good — that's the only Criterion!

Clearly, for any design an ultimate Criterion is *does it work?* but usually this can be decomposed into sub-Criteria which serve that goal. The designer *did* make Criteria such as *consistency, flexibility, non-repetition of data,* and *reduce real time calculation* explicit during Sessions 2 and 3, but the difficulty in structuring the deliberation process into subissues meant that Criteria like these remained useful only at a global level of application, rather than for alternatives to subproblems within the design space.

7.2.2.3 'Strategic' versus 'evolutionary' design

The contrast between the two kinds of problem solving typified by Session 1, and Sessions 2 and 3, was noted by the designer himself. Towards the end of Session 1, he began to sketch a message-passing structure, in order to "see how I would approach this if I was doing the low level stuff." At this stage, he did not know what he would be doing in Sessions 2 and 3; however, it turned out to be exactly the sort of design problem which he did tackle, as borne out by the similarity of the following Session 1 comment with his comments from subsequent sessions:

> [Sess.1] ... it's more a case of you don't really see the other [Option] ... until you test it. At this stage (indicates first Option) you see that, and it's all you can think of doing – the first Option if you like. And it's not an either/or situation.. this (i.e. Session 1) *was* a case where there were decisions to be made, but most of the time it's more subtle than that... there's one thing made up of lots of interlocking bits, and you only find out what's wrong by testing it.
>
> You do have Criteria of course – it's got to enable certain flexibilities and possibilities to happen, but you don't then say 'well that one was negative for that, so we won't do that protocol, what's this other one?' It's really that the second Option only developed out of trying the first one, and may be based quite a lot on the first one as well.

Some of his comments on the different kinds of design are illuminating:

> [Sess. 1] ...whereas if it's an actual 'decision' that you made: ' we're going to use a prestored data packet' ... it's a *decision.* The implementation is to be something else. Then if someone says why did you decide to do that, then that [DR] is a good thing – you can go back and say well if we did it the other way, etc.
>
> I think almost there's two different.. the activity I was doing here was different from what I was doing yesterday – the actual low level structures [i.e. in work prior to Session 1]. This is more about big decisions – policies – whereas this (refers to own notes) is about implementation. So this [DR] is certainly very useful for strategies.

> [Sess. 2] [referring to Options on the Session 1 DR] you can see the Criteria – *low processor calculation required.* I could just tell that without knowing what processors were going in where, etc. Yes so it was a different type of decision making. I find this diagram [the DR] very useful in order to be able to justify to other people why we've gone for this.

> [Sess. 1] This is a policy decision: we do it by storing data in a second unit generator and then reconnecting it.
>
> ... [referring to using QOC] ... it was quite tricky to think in that way then [for Session 1] but it was useful... but I think I'd find it almost impossible to wrench myself into thinking in that way (for low level design); it would be unnatural almost, 'cos it's not that kind of path that you take when you're doing this sort of thing.

7.2.3 CONCLUSION: WHEN WAS QOC USEFUL?

The designer used two separate analogies in the course of describing how he was working:

> ... have you ever watched someone design a circuit board?
>
> It's more ... like painting a picture of something, and you ask why did you put the trees there and not there? There are so many Criteria and they all interrelate.

These images aptly sum up those design situations where representation of DR is extremely difficult. The nature of such work involves balancing so many decisions and dependencies that they cannot all be systematically articulated without causing disruption. Moreover, such decisions are not the kind which are likely to be queried by domain experts, because it is routine design; whilst demanding, it has value only in that it implements higher level decisions which shape the design – it is these decisions for which DR notations based around argumentation would appear to be best suited and most valuable.

Chapter 8

Case Study 2: Developing and Evaluating a Revised QOC Tutorial

8.1 The revised QOC tutorial

This case study was effectively the first step in the development of a methodology for structuring design within the design space analysis approach to DR. The aim was to evaluate a process model and heuristics for designing with QOC, taught in a revised version of the QOC tutorial developed in Study 3. The tutorial, presented to designers in the research and development division of a software company, introduced an informal 'methodology' for developing QOC analyses and representations—what shall be referred to as the 'QOC process model.' This model (and by implication the tutorial) were evaluated through two design exercises, in which two design teams each tackled two design problems.[1]

This report first describes the reasons for extending the Study 3 tutorial in certain areas. The results of the exercises used to evaluate the process model are then reported, and prospects for future work discussed. As the data in this case study were collected relatively late in this research project, it has not been possible to analyse the design activities in as much detail as one would have wished ideally. Conclusions which need to be treated as pointers to future work are therefore specifically noted.

8.1.1 The designers studied

Six designers were presented with the tutorial as a group, and tackled the design exercises in two teams who were used to working together, each of three designers. Professional experience and areas of expertise are shown below for each team (as explained later, one is labelled NetGroup, and the other FileGroup):

---

1 Ideas for the QOC tutorial and process model were developed in collaboration with Allan MacLean and Victoria Bellotti; the final content of the tutorial was decided by Allan MacLean, who also presented it. The tutorial materials are compiled in MacLean et al (1991b), a summary of which can be found in Appendix 10. The video-data was collected in conjunction with Victoria Bellotti, and David Elworthy, who also transcribed the videos.

Expertise	Professional experience
NetGroup	
software engineering	9
knowledge-based systems	4
AI/knowledge-based systems	4
FileGroup	
mathematical aspects to software engineering	3
neural nets	2
genetic algorithms/neural nets/knowledge-based systems	5

Table 8.1: Designers studied in Case Study 2

8.1.2 STRUCTURE AND TIMING

The data for this case study were collected in one day, the tutorial taking three hours in the morning, with four hours in the afternoon for the design exercises and debriefing. The design problems and a summary of the tutorial materials can be found in Appendices 10-12. The structure and timing of the tutorial and exercises are summarised in Table 8.2.

9:30	Introduction
	(background to design rationale, and uses of QOC)
	Crib-Sheet (used for reference during design exercises)
	Notational conventions
	How to design with QOC
	Some common problems
	Example: Scroll bar
	(illustrating basic QOC analysis)
11:00	*Break*
11:15	Example: Basic ATM
	Example: ATM Rough DRs
	(illustrating the design process, QOC refinement; heuristics)
	Process overview
	(details of each Phase in QOC process model)
	Exercise: ATM 1
	Exercise: ATM 2
	(small exercises performed individually, and then discussed)
	Questionnaire
	(feedback on tutorial, familiarity with QOC notation, and confidence in QOC as a notation and approach)
12:45	*Break*
13:45	Design exercise: CSCW user interface problem
	(artificially devised problem)
	(both exercises performed by two existing design teams of three)
	Questionnaire
	(answer same questions)
15:15	*Break*
15:30	Design exercise: Your own problem
	(designers tackle problem currently faced in own work)
	Questionnaire
	(as before, plus additional questions on experiences with QOC)
16:45	Debriefing & Discussion
17:30	End

Table 8.2: Structure and timing of the revised QOC tutorial.

During the design exercises, the designers were able to refer back to a three-page ‘crib-sheet’ (Appendix 11) which summarised the main points of the tutorial: QOC notational conventions, the five phases of the QOC process model, eight heuristics for developing QOC structures, and a list of common problems and hints (e.g. ‘getting stuck’ with QOC, and emphasising the need to find good Questions).

8.1.3 CONTENT OF REVISED TUTORIAL

This study was conducted when much of the analysis from Studies 2 and 3 was well advanced; consequently, the demands of conducting good QOC analyses were becoming clearer. As a result it was decided that the revised tutorial should emphasise several aspects of QOC authoring which the Study 3 tutorial did not address: the freedom inherent in the notation to produce poor representations, the role of rough, evolving QOC representations which necessarily precede coherent analyses, and the non-linear nature of the QOC authoring process. These are described in turn.

8.1.3.1 ‘Good’ and ‘bad’ QOC representations

Studies 2 and 3 had shown that it is easy to produce QOC representations which reflect poor analysis of the problem, and ongoing QOC research was also addressing similar issues (Bellotti et al, 1991). As Study 3’s tutorial did not devote much time to these problems, a goal of the revised tutorial was to give the problem a much higher profile. One way in which this was done was via several examples of ‘poor’ QOC due to unfocussed Questions, poorly differentiated Options, and common confusions between QOC node types.

A second approach to the problem was to support the designers with heuristics to facilitate well-structured analyses. MacLean et al (1991) proposed nine heuristics, five of which were formulated to help to “locally expand the notation”, the remaining four being “aimed at dealing with larger patterns in the notation.” For the tutorial, three more local heuristics were added to form eight simple guides (Table 8.3). The global heuristics were not presented, partly because they are most relevant for large structures, but also to avoid overloading the designers with too many rules.

In the tutorial, the heuristics were illustrated in a step-by-step development of a QOC space for scroll-bar design. They were presented in the role of prompts to which the designers could turn when stuck in their analyses, and were reproduced in the crib-sheet of reminders about QOC for use in the design exercises. Developments to the concept of heuristics are considered in the general discussion [§10.5].

LOCAL HEURISTICS

- H1. Use *Options* to generate *Questions*
- H2. Use *Questions* to generate *Options*
- H3. Use *Criteria* to generate *Options**
- H4. Use *Options* to generate *Criteria**
- H5. Consider extreme *Options**
- H6. Consider distinctive *Options*
- H7. Represent positive and negative *Assessments*
- H8. Overcome negative *Assessments*, but maintain positive ones

(GLOBAL HEURISTICS)

1. Identify *Options* which generate dependencies
2. Look for novel combinations of *Options*
3. Design to a set of *Criteria*
4. Search for generic *Questions*

*Table 8.3: Heuristics for QOC analysis. Only the eight local heuristics were used in the tutorial. (*Heuristics added for the tutorial; the rest from MacLean et al, 1991).*

8.1.3.2 Rough QOC

Rough QOC is a concept developed in the light of Study 3, following the analyses of QOC authoring as a form of writing [§1.3.2], and in terms of cognitive dimensions [§10.6]. Conceptual parallels with knowledge elicitation also exist [§10.5.1]. The underlying premise is that it is often impossible for designers to express rational arguments directly in terms of the semi-formal framework offered by QOC and other DR notations. Consequently, less constrained notational forms are needed in order to bridge the gap between initial formulation and recording of ideas, and their subsequent rationalisation.

A new feature in the tutorial was its emphasis on rough QOC as a stage in the representational process in its own right. During the design exercises, an A3 sheet was provided expressly for the purposes of drawing up ‘first-pass’ rough QOC, making incomplete DRs ‘legitimate’ structures to work with. The sheet consisted of three columns for Questions, Options and Criteria, and a working area at the bottom for ideas which could not be immediately classified, plus other notes (Figure 8.1).

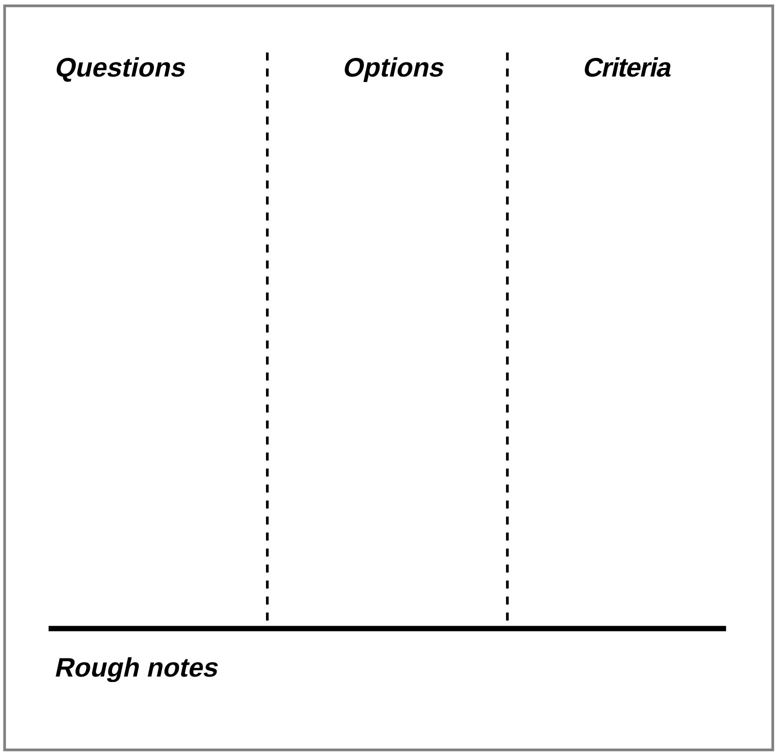


Figure 8.1: A simple form for recording rough QOC

Rough QOC represented Phase 2 of the five phase process model presented to the designers, described next.

8.1.3.3 A QOC design process model

In designing this tutorial, it was clear from the available research that designers might benefit from some clear guidance as to how to use QOC. This is particularly important if designers are to develop 'good' QOC analyses, which rarely emerge in the first-pass representation. As a result, a method for DSA was devised, shown schematically in Figure 8.2, elaborated in Table 8.4 (both taken from the tutorial materials).

Whilst initially this model appears to prescribe a linear, top-down procedure, the tutorial emphasised that designers should feel free to switch between phases, since the limitations of more rigidly top-down software design models were openly recognised.

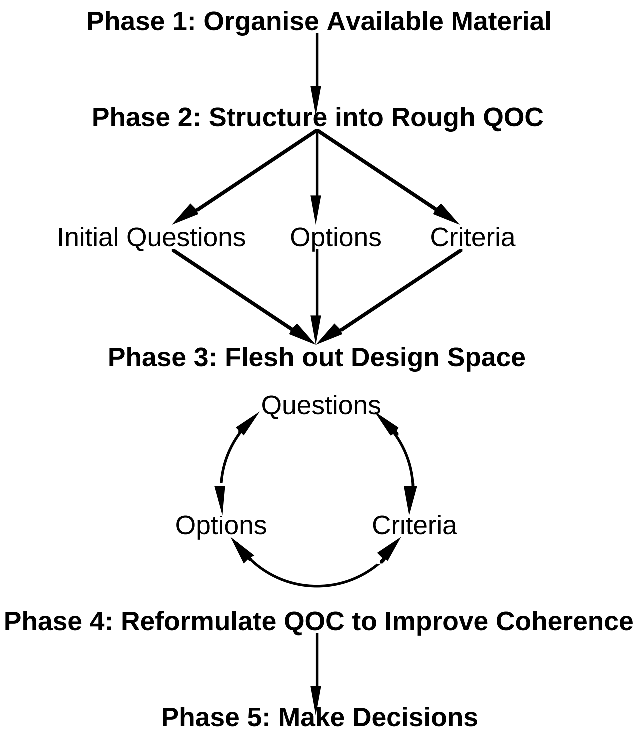


Figure 8.2: The process model for QOC DSA, presented in the revised tutorial.

Phase 1:	**Organise available material**
Tasks:	Get relevant information down Get a feel for the main issues Work out what information provided is relevant (& classify as Q,O C if possible)
Phase 2:	**Structure material into rough QOC**
Tasks:	Structure and make sense of the information available Find good Questions
Phase 3:	**Flesh out design space**
Tasks:	Use current understanding of design to help generate new ideas. Generate new Options Generate new Criteria
Phase 4:	**Reformulate design space to tidy it up.**
Tasks:	Tidy up description and make it more coherent Reword Q, O, C if necessary Reformulate Questions (and reorganise O, C) to improve decomposition
Phase 5:	**Make design decisions**
Tasks:	Evaluate and select Options (i.e. draw boxes around decisions) (Use Criteria to evaluate Options) (The level of detail represented may not include all relevant information – you may need to consider the importance of Criteria, or their Assessment, or interdependencies with other parts of the space.)

Table 8.4: Details of the phases in the revised QOC process model

8.1.3.4 Structure to assist use of the process model

The goal of this case study was to assess the above model of a QOC-oriented design process. However, there are practical problems which inevitably arise when a group changes its method of working, namely *who* does *what, when* and *how,* and there was the danger that these factors might swamp other behaviour of interest. In order to assist the designers in following the method, approximate timings for each phase were suggested, and a role for each designer, shown in Table 8.5.

Timing:

Phase 1:
Identify issues/ initial crude classification of given material........... 10 mins

Phase 2:
Structure into rough QOC .... 10 mins

Phase 3:
Flesh out design space .... 15 mins

Phase 4:
Tidy up QOC .... 15 mins

Phase 5:
Make design decisions .... 10 mins

Roles:

Scribe: Records QOC

Timekeeper/Crib-minder: Monitors group progress through the phases of the method, and checks the crib-sheet for hints on how to proceed.

Problem minder: Monitors problem requirements (and for the artificial problem, information in the problem statement of potential use).

Table 8.5: Suggested timings for process model phases, and roles for designers.

To summarise so far, an explicit model for designing with QOC was presented, which the designers were asked to follow during the subsequent design exercises. To assist them, specific roles were assigned within each team, approximate timings for each phase were suggested, and a crib-sheet provided. It was hoped that with this structure in place, consequent behaviour would be less affected by initial learning problems (e.g. forgetting the notation or heuristics; not having enough time to refine the QOC representation), and as a result would reflect more on the efficacy of the process model and the approach itself.

8.1.4 PROBLEMS USED IN DESIGN EXERCISES

The designers tackled two one-hour exercises, each exercise being performed by each team. The first exercise used a problem provided by the QOC tutors, and in the second, designers tackled a problem selected from an existing project in which they were engaged (which had been approved by the experimenters as suitable).

8.1.4.1 The ‘P&O’ problem

The ‘People and Objects’ (P&O) problem (Appendix 12) was based on an actual prototype multimedia environment which linked users together so that they could access documents of different kinds. The design task focussed on evaluating and improving two alternative user interfaces designed to show which users were using which documents. The designers were asked to use QOC to evaluate the designs, improve them if possible, and summarise their conclusions. This problem was chosen for the design exercise because it was sufficiently constrained for the one hour available, and provided numerous alternatives and criteria to consider.

8.1.4.2 NetGroup problem: Browsing a graphical network

One design team (which shall be referred to as NetGroup) were concerned with designing information access to a graphical network which would be larger than one screen. (The network represented dependencies between chemical compounds as part of a truth maintenance system). The network nodes had the facility to serve as hypertext link buttons which could be selected to display their contents, or new views of the network for instance. The issues of concern were how to display the global structure of the network, with mechanisms for accessing more detailed views.

8.1.4.3 FileGroup problem: Optimising the format for data files

The second team (FileGroup) were concerned with designing the optimal file format for storing data files. Their declared requirements were to minimise disk space, to be able to check the consistency of files, and to store sufficient information within a data file (the specific details of this work are proprietary, but the essence of the problem is sufficient for present purposes). The designers had specified two clear issues to resolve in the exercise, which were to decide on a file format, and to design the file header which carried the information needed to interpret the data in the file.

8.2 Results from design exercises

Transcriptions of the video data were prepared, similar to those used in Study 3, recording the interweaving of sketches, notes, and QOC during discussion. In the following sections, patterns of QOC use are reported, and designers' activities for each problem are described in relation to the process model. Possible explanations, and implications for the model are considered in the general discussion. Use of the heuristics is discussed in a single section covering all three design problems [§8.2.3], as results differed only slightly across problems and groups.

8.2.1 Replicating QOC authoring patterns from previous studies

This section briefly reports on observations of the QOC authoring process, of which seven characteristics in particular confirmed the picture of QOC authoring emerging from the other studies.

(i) The basic QOC authoring task of *classifying an idea* led to the by now familiar debates as to whether something was, for instance, an Option or Question [Study 3 §6.2.2].

(ii) The task of *naming* nodes succinctly also led to regular *renaming* as the nature of entities became clearer in relation to the rest of the rationale [Study 3 §6.2.3].

(iii) Another aspect of authoring which arose in Study 3 was that of the *level of abstraction* of the DR [§6.2.3]. As FileGroup began to record rough ideas as QOC on the column-sheet, they were conscious that they could 'go in' at the wrong level, either too general or specific. To begin with, several specific solutions were suggested, but then summarised more generally by the scribe; this in turn caused concern that they were being too vague:

D	I have trouble knowing at what level one should pitch in with this, but alphabetic sorting of people (indicates Options column) is an Option we have
I	yes (writes ALPHABETIC SORTING OF PEOPLE)
D	another Option is to group them by, say, office, perhaps some of them work in the same building, and know that...
I	indeed. Well, grouping...Option... well, perhaps the sort of Options at this point, we shouldn't talk about sorting people, we should talk about methods of grouping people... (deletes first Option , and writes METHODS OF GROUPING PEOPLE & OBJECTS)
D	fine, ok, yeah
	(...)
G	and then you've got methods of finding object owners, basically
I	(writes METHODS OF FINDING OBJECT OWNERS) These are sort of getting high level.
D	yeah, we don't want to get too high level, because at the end of the day we want to design something, so we need some Options enumerated
G	yeah, we can do that.
I	yeah, ok. I'll put stuff...

When they had developed the design space further, implementations of the general Option METHODS OF GROUPING PEOPLE & OBJECTS were enumerated explicitly.

(iv) The need arose to *weight Criteria and Assessments*, as in Study 3 [§6.2.4, §6.2.5] and Case Study 1 [§7.2.1.3]. Thus, in one instance, FileGroup generated an Option which despite being both *slower* and more *complex* than another, was chosen because *flexibility* was critical. As an example of Assessment weighting, NetGroup used what they called "long dashes" to indicate that an Assessment was less negative than others in that Question.

(v) *Option decomposition/refinement* occurred via consequent Questions. For example, NetGroup asked an initial Question (*which representations to use?*), and later pursued the decision (*graph-like*) with a series of more detailed Questions (*what type of graph?, what method of navigation?*).

(vi) *Meta-argumentation* [§4.2.6.1] refers to debate about the validity of entities or relations used in the DR structure, for instance, the validity of a Criterion or Question. A clear example of meta-argumentation occurred in FileGroup's discussion of P&O, in choosing between mouse vs. keyboard specification of names. Designer *'G'* challenged the Assessment that TYPE NAME was faster, but gave way in response to the counter-argument:

I	TYPE NAME wins on SPEED OF SELECTION (adds supports link), loses out on ERRORS (links in), because

G	do you think it's good for SPEED OF SELECTION? I would have thought it's quicker to go click, click than it is to type, unless you're in a continuous stream of typing, but remember looking at what this is (indicates one of screen displays)...
I	providing a person's on screen, it's quicker...
D	the problem is...
I	if you have to scroll. Once you have to scroll, you've lost.
D	yeah
G	yeah - once you have to scroll you've lost, yeah.

This class of argument can be represented in QOC by using *Argument* nodes (Figure 8.3), as outlined by MacLean et al (1991). (However, this was not part of the tutorial). QOC Arguments are a generalisation over Lee's DRL constructs for meta-arguments, namely Claims, Questions, and several different link types. A DRL representation would look very similar, except the Arguments would be Claims; however, unlike DRL, QOC as it currently stands would not be able to express meta-questions, e.g. *how often would we expect icons to be off-screen?*

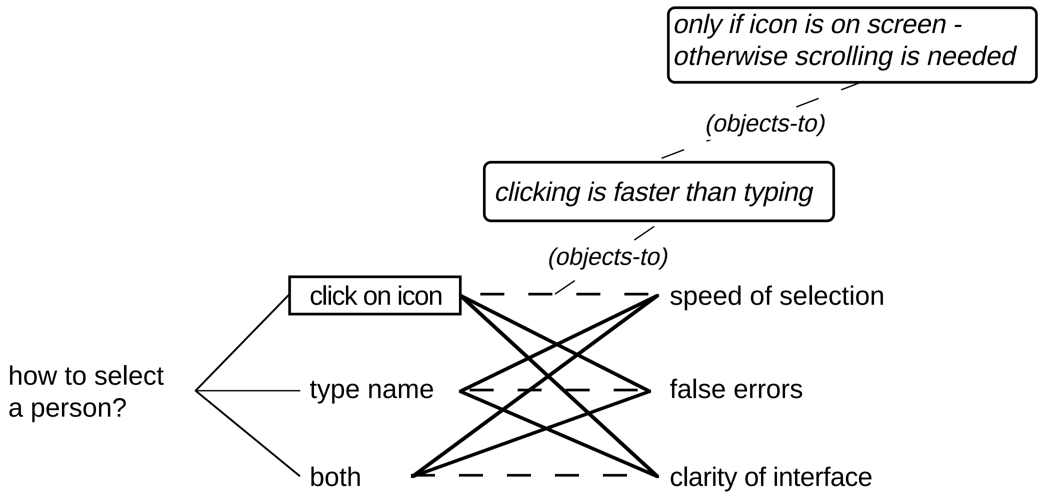


Figure 8.3: Meta-argumentation in QOC: using Arguments to contest an Assessment (from FileGroup)

The question of how useful this level of argumentation will be generally is still very much an open issue [§3.3.1.3; §10.8]. However, in this particular example, it is useful to see the backing arguments to what might otherwise seem a counterintuitive Assessment.

(vii) The last characteristic of interest, is the possibility of *combining Options.* This issue arose earlier in Study 3 [§6.3.1.1]. There it was observed that QOC and other DR notations assume a model of design based on choosing between competing alternatives, rather than on combining different elements in different ways.[2] NetGroup noted that for an Option to use *multiple levels of network abstraction,* the next Question would be *what are those levels?* They faced the same problem as the other designers:

P	it [QOC] seems very good at giving you disjunctions and...
G	choices between...

2 In two other studies, 'elements' were (i) graphic design elements, and (ii) properties of Smalltalk data structures – in both cases they were constantly being modified.

P	...choices between things, but it's not very good at expressing combinations of things.
G	yeah.
P	you know, I mean here (indicates possible Option) when the next Question is 'what are those levels?' you don't want to go into... you don't really want to say 'this, this and this' or 'this, this and this' – we just want to say 'it's going to be this, this, this and this.'

In this example, the question is whether each abstraction level, or whether each *set* of levels should be an Option. Yet another way (not considered by the designers) is to have one level per Option, and select several Options. Ultimately, the answer may be simple. In some kinds of design, QOC may not be suited to the mode of working, as in Case Study 1. However, where laying out the different elements is deemed useful, the notation should permit the designer freedom to represent the alternatives in any of the above ways, as the context dictates. Examples of common representational problems such as these could be covered in a more advanced QOC tutorial.

8.2.2 USE OF THE QOC PROCESS MODEL

Attention now turns to the mapping between the process model, and the teams' use of QOC during the two design exercises.

8.2.2.1 Design exercise 1: P&O problem

8.2.2.1.1 NetGroup's use of the model

NetGroup developed a relatively legible and detailed QOC analysis (six Questions with Options and Criteria) on their rough QOC sheet , so, although not explicitly declared, they were already elaborating the design space (Phase 3). Consequently, when they felt that they should move to a 'proper' QOC representation on one of the sheets provided, they decided instead to continue to work with the rough QOC sheet, rather than rewrite it. They went on to add Assessments and new Criteria to the existing Questions, plus two new Questions.

It may be that had they rewritten the first-pass QOC, more restructuring would have occurred. It is also possible, however, that it might have been treated as a simple transcription task. A third, most likely interpretation, is that they were able to represent a relatively well-structured analysis initially, so that there was no need to restructure—their Questions targetted relevant issues (avoiding Yes/No Questions), and they posed consequent Questions where appropriate.

8.2.2.1.2 FileGroup's use of the model

FileGroup used the columned rough QOC sheet for initial discussion (18 minutes) during which they recorded the major goals of the design solution (e.g. Criteria like good communications; clarity of display), several general classes of Option (e.g. methods of finding object's owner), and a number of Questions for the key issues. They then structured them as QOC (Phase 2), adding new elements as they arose in discussion (Phase 3). However, as with NetGroup, reformulation (Phase 4) did not really take place, except in renaming of nodes to modify or clarify their meaning in relation to other nodes; there was no restructuring of Questions.

Finally, both groups deferred decision-making to the end (Phase 5) when the design space had been laid out. In Study 3, there was tendency to make decisions before moving on to the next Question, unless there was a dependency which needed first resolving. In this respect, therefore, the process model was effective in its goal of delaying commitment within the design space. Furthermore, both groups commented that the process of boxing Options in a systematically laid out design space was useful.

In sum, the process model was followed quite closely in exercise 1, with the exception of Phase 4. There was very little explicit reflection on the first-pass QOC representation, and apart from modifications to names, there was very little reformulation (Phase 4) before final decision making. Both groups assigned roles as suggested, and the time-keepers ensured that they stepped through the model's phases.

8.2.2.1.3 Comparing the DRs generated for the P&O problem

The QOCs from the two groups are reproduced in Appendix 13. Comparison shows that they addressed a similar number of issues (Netgroup eight, FileGroup nine), at a similar level of detail, and topic. This is perhaps not surprising, given that the P&O problem was chosen as a vehicle for studying QOC use, and both the designers' and the hypothetical users' tasks were well-defined. However, it does also attest to a certain level of proficiency with the notation.

8.2.2.2 Design exercise 2: Designers' own problems

As described above [§8.1.4], the two design groups tackled very different kinds of problems in the second design exercise. *NetGroup* were concerned with user interface issues to do with network display, whilst *FileGroup* worked on a 'lower level' aspect of their project, relating to the optimal format of data files.

8.2.2.2.1 NetGroup's use of the model

It was found that NetGroup followed the process model closely. They spent longer than before on recording key issues as unclassified notes, classified and elaborated several as QOC-columns (Phases 1, 2 and 3), and developed a more elaborate sequence of Questions which explored further an initial decision to display graphical representations (which could be seen as Phase 4 development of the QOC). They made their decisions last of all, as the model suggests (Phase 5).

8.2.2.2.2 FileGroup's use of the model

The designers began with rough notes on the QOC-column sheet, some of which were classified, others left as notes; at this stage there were some brief Assessments showing the main strength of each Option (Phases 1 and 2). After just under 10 minutes' discussion, they felt that they had isolated the first main Question, with Options and Criteria, and proceeded to elaborate the space around the Options on a new sheet (Phase 3). There was much quick discussion of several issues which was not recorded, but when another important issue arose it was made a Question and quickly resolved. There was then an intense period of discussion during which they recorded only a few notes and one more Criterion, until the designer responsible for monitoring phases observed that they were not using QOC much. On reflecting on their discussions, they realised that they had in fact been exploring the relative importance of the Criteria to their first Question, and decided to add weightings in accordance with the conclusions they had just reached. This enabled them to resolve the first of the two issues which they had set themselves.

For the second issue (contents of file header information) the designers returned to a rough list of requirements, to which they added regularly. To resolve the problem, they recorded six Questions via cycles of discussion and 'bursts' of recording QOC. These brief representational activities consisted of recording an issue as a Question (two of them simple Yes/No questions), rapidly noting Options, Criteria and Assessments, and making a decision.

Although the process model does not restrict designers to strictly stepping through the phases, one of its aims was to introduce some method to the translation and structuring of ideas as QOC, that is, *in general*, the designer should 1–generate and translate ideas, 2–structure, 3–elaborate, 4–restructure, and finally, 5–make decisions. FileGroup's mode of working exemplified a process of 1,2,5~1,2,5~1,2,5, that is, variable amounts of discussion interspersed with bursts of QOC activity. A graphical representation of activity streams in the sessions such as meeting plots (Olson and Olson, 1991) would highlight this pattern of working. Although there was not time to generate such representations in the present study, Elworthy (1991) in his analysis of this data uses them to track the movement between Questions over the session.

It is interesting to observe rough, first-pass QOC flowing so naturally from discussion, since in Studies 2 and 3 this occurred less frequently. The key seems to be that FileGroup's QOC was grounded in 'concrete' argumentation about problems. That is, a question with its alternatives and trade-offs *had been discussed*, so that the ideas flowed quickly when actually recording QOC (in one case the problem had already been resolved before moving to QOC). This strategy avoided the problem of 'stalling' which occurs when trying to generate Options or Criteria simply because the formalism demands it. In previous studies this had disrupted the flow of discussion, and had been a source of complaint.

Elaboration and reformulation (Phases 3 and 4) occurred only to a very limited extent (e.g. adding a Criterion to an earlier Question; changing an Option name when its meaning became clearer).

8.2.3 EVALUATING THE QOC HEURISTICS

The eight heuristics provided were described earlier [§8.1.3.1]. The key result from the exercises is that none of the heuristics was consciously applied by either team as a group strategy for developing the design space. The transcripts suggest that they were most probably forgotten, as they were not mentioned.

It is not, however, difficult to identify instances where the designers' discussions and use of QOC could be aptly *described* by one or more of the heuristics. For example, it is impossible to represent a simple QOC structure without implicitly following several of the syntax oriented heuristics H1-4, and H7 (shown below), so in this limited sense, they were used automatically by both groups:

H1. Use *Options* to generate *Questions*
H2. Use *Questions* to generate *Options*
H3. Use *Criteria* to generate *Options*
H4. Use *Options* to generate *Criteria*
H6. Consider distinctive *Options*
H7. Represent positive and negative *Assessments*

H6 is not as rudimentary as the others, as it is quite possible to use Options which are indistinct in some way (e.g. by confusing the issues to which they respond; by inadvertently combining alternatives; by expressing essentially the same alternative in two ways – observed in Study 3 [§6.2.5]). In the present study, the majority of Questions generated distinctive Options, rather than close variants of a single Option. The tutorial may have assisted here, as it illustrated the role of this heuristic; however, the Study 3 designers were also able to generate distinctive Options, which makes it difficult to draw strong conclusions on the basis of this initial trial of the tutorial.

Neither of the remaining heuristics (below) were used as explicit strategies for exploring the design space:

H5. Consider extreme *Options*
H8. Overcome negative *Assessments*, and maintain positive

Whilst H1-4 and H7 focus the designer closely on the specific syntax of QOC, H5 and H8 can be viewed as recommending broader approaches to tackling problems. H5 is an unusual way in which to approach a problem, so it is not surprising that it was not used as a matter of course. Whilst extreme Options were considered on occasions, this was not as a result of any explicit decision to brainstorm over the problem in the manner suggested by the heuristic (for instance NetGroup recorded an Option whilst making it clear that they thought it obviously no good).

H8 focusses the designer on pursuing negative Assessments to Options, in order to minimise, or even eliminate their disadvantages by, for instance, thinking of better Options, reformulating the Question, or asking a consequent Question whose Options overcome the weakness. NetGroup overcame weak Options with better ones (without actually articulating the heuristic); FileGroup did recognise H8 (retrospectively), citing an example where overcoming a weakness had led to an improvement to the P&O interface (the difficulty of searching a large icon set led to the decision to use *viewing modes* such as alphabetically, or by type, date, and so forth). However, the heuristic is again *describing* what they did from an analyst's perspective, as opposed to shaping or *prescribing* designers' action during design.

8.2.3.1 Conclusions on the teaching and use of QOC heuristics

In the normal sense of the word, a heuristic is 'good' if it can be appropriately applied, in this context, as a strategy for designing with QOC. Only FileGroup made explicit reference to the heuristics, and then only when they were remembered towards the end of the P&O problem. FileGroup judged that they had been using all of them "as a matter of course" (which was true, to the extent described above).

More detailed analysis of the transcripts would reveal which heuristics were being used implicitly, but the wider question remains: does the fact that heuristics have 'face validity' to the extent that they are able to *describe* design activity mean that they are good heuristics to teach for *shaping* design? There are several arguments which one might draw upon here.

If a heuristic is so embedded in the natural activity which it is meant to support that it occurs as a matter of course (which was the case with all but H5 and H8), it may be more appropriate to 'relegate' its status from *heuristic* to that of a *rudimentary task* in using QOC. Alternately, if problem solving or QOC recording breaks down, having relatively 'intuitive' strategies explicitly set out for reference could be useful (this was one envisaged scenario of use when the heuristics were devised). The converse could also be argued however, namely, that the heuristic should introduce something fresh into the problem solving, rather than encapsulate elements of what designers already do. A response to this might be that designers will never use what they judge to be too extreme an approach to design, and that encapsulating naturalistic behaviour as heuristics serves to ensure greater rigour in design, given the weaknesses in design activity at both the cognitive and group levels (Guindon et al, 1987; Olson and Olson, 1991; in press).

To conclude, experiences from the teaching and evaluation of this version of the tutorial suggest that in contrast to heuristics which are used as a matter of course, more prescriptive ones may be neglected because the designers' efforts are engaged simply in using QOC whilst solving the problem. The overhead leaves no room for more advanced strategies (e.g. H5), or the *global* heuristics, such as *identify generic Questions* or *dependencies*. It may only be possible to use these once QOC is more familiar, which indicates that these more powerful heuristics could form part of an 'advanced' tutorial for more experienced QOC users, for example, designers who have designed regularly with QOC for 1-2 months. The prospect for a different form of QOC heuristic (analogous to 'production rules' for making moves in QOC) is considered in the general discussion [§10.5.3].

8.2.4 Designers' reactions to QOC

It was noted in the above sections that neither design group encountered serious difficulties in using QOC, at least not to the extent that they were unable to rationalise ideas within the notation's framework. Although they did not engage in much restructuring and 'cleaning up' of their QOCs (Phase 4), the representations were relatively well-structured. The evidence is that QOC can be learnt and used at least in a 'minimalist' fashion without excessive disruption (that is, for recording decisions, without much restructuring).

However, comments during and after each exercise revealed that particularly within FileGroup, the prevailing attitude to QOC was not favourable. This was largely because it was felt that using QOC held back their normally fast-flowing design meetings. Whilst some of the criticisms expressed were in part simply a function of the group's personalities, the summary of points made (below) raises a number of issues pertinent to QOC's usability (parentheses indicate where NetGroup made similar comments):

- FileGroup was used to freely brainstorming as a team – they felt that QOC inhibited this;
- "the notation can make or break a method"—QOC as a notation was inadequate and cumbersome (NetGroup also);

- a lot of DR seemed to be generated even for a small problem;
- there was a perception that QOC's design model was to step through each decision by enumerating possibilities at each stage, which was felt to labour things, and was too constraining (NetGroup also);
- the scribe got 'left out' as discussion moved on whilst he was recording the last decision;
- it was "hard to see how a design hangs together" when decisions are distributed over a large area—there needs to be some way of pulling decisions together;
- it was felt that the key ideas they had generated had not arisen from using QOC – "it was intuition." They worried that formalisation might lead to such insights being missed;
- there were points at which the way forward seemed to be clear, but they felt they had to wait for the DR to 'catch up';
- similarly, there were times when a good decision had been made, but the line of reasoning had moved far ahead of the DR—the structure was out of date (NetGroup also commented on this: "we've sort of arrived at all this solution, but we haven't really used the... [QOC method]");
- the heuristics were not found to be useful; there was unanimous agreement that they used them "automatically", and "as a matter of course." *Use extreme Options* was considered to be an odd way to design.

There were however also positive comments:

- rationalising design is something that is not done enough.
- keeping a purely logical (as opposed to chronological) record is a good idea;
- noting decisions as one designs is an excellent idea, and valuable for recapping and consolidating ideas, in particular the process of systematically marking Decisions at the end (NetGroup also). (This is consistent with the need to delay commitment as long as possible – Goel and Pirolli, 1989)

Compared to FileGroup, NetGroup were less critical, as QOC appeared to be more useful in their own problem. Possible reasons for this are considered shortly. The above problems fall under two headings which have emerged as themes in this thesis:

(i) the adequacy of QOC as a representation for DR, and
(ii) how to record QOC, yet maintain the dynamic group interaction typical of rapidly developing design ideas.

Designers' use of and subjective reactions to QOC can be understood as a function of at least the following three variables:

- *the QOC notation and support environment*: the notation's expressiveness (domain coverage) and perspicuity (how salient important elements are). If there is a tool, the user interface and computational support for the designer are clearly important.
- *the design problem*: is the structure of the problem ill- or well-defined? Can a tried and tested method be recruited, or is a lot of exploration of the problem space necessary?
- *the QOC process model*: independent of the problem or notation, the process model can be modified to better support the integration of QOC with design; the better understood the interaction between the notation, and mode of working, the more precisely the process model can be shaped to support (and hopefully) augment problem solving.

Each of the two classes of difficulty noted above can be considered in terms of these three variables. The first complaint, QOC's sufficiency as a formalism, is clearly a function of the first variable. The power of the DR notation and environment has been a recurrent theme in these studies, and the fact that it recurs again in this study serves to highlight its importance. Notational design is considered elsewhere in relation to cognitive dimensions [§10.6]. Study 1 focussed largely on the management of large DR structures, whilst expressive properties of QOC like *meta-argumentation*, and the *weighting* of Criteria and Assessments were addressed in Studies 2 and 3.

The second difficulty, maintaining a smooth integration of QOC with design activity, is essentially the prime motivation for studying the characteristics of QOC authoring in detail, and has possible relations to all three variables. Leaving aside notational design at present, the variable of *problem structure* is considered in the remainder of this chapter. A perspective on the *process model* is presented in the general discussion [§10.4], drawing on work by Tang (1989) which offers insights into the use of representations in group conceptual design.

8.2.5 PROBLEM STRUCTURE

Whilst in a different domain, NetGroup's problem was similar to P&O in the kind of analysis required. At one level, the problem content was similar, both involving user interface design focussing on optimal use of display space, and ease of locating information. More important, however, is the problem structure. Compared to the file format problem, both P&O and the NetGroup problem were relatively open-ended. This is in part a function of the domain, in that user interface problems have no ultimately 'correct' solution, whereas alternative methods for storing data can be judged more strictly. This point emerged in the final debriefing, in which FileGroup commented on the difference between the P&O and file-format problems:

> G: In our last problem [i.e. file-format], it was actually a problem about how you store things which has real physical constraints and measurable results from doing something in a very real sense. So we could sit there and say, well, we could do it in compressed binary or whatever, and then say, we won't do that because it gives you this factor of something, whereas in the P&O one we were saying, well this would be nicer to look at, more accessible, in some more, some less quantifiable way...

[Our] final problem ... was much more hard and fast and much better defined in some sense...
I: It wasn't so much selecting options or even selecting criteria, as measuring the criteria; we spent a lot of time deciding that... deciding the real critical trade-offs was actually the whole point.

Given that P&O was chosen for the tutorial with QOC in mind, it might be predicted that QOC would prove useful for NetGroup's problem. In terms of the structuredness of the problems, this is consistent with the original work by Rittel (1972), who developed the argumentative model of design in response to the limitations of existing approaches to ill-structured, 'wicked' problems [§1.3.1.3]. Compared to NetGroup, FileGroup were tackling a relatively well-defined software problem (i.e. not 'wicked'); the problem was more focussed in scope, and better articulated at the start of the session. Whilst this made it 'easier' in some senses, it also reduced the utility of DR in shaping the problem, relegating QOC to the role of passive record (a distinction drawn out later in relation to Tang's framework [§10.4]). In contrast, NetGroup were working with the less certain principles of interface design, and a poorly articulated notion of the task for which they were designing. Consequently, there was more scope for exploring the problem and solution spaces. A similar contrast was made in Case Study 1, between the different modes of working engendered by the kinds of design problem tackled.

8.2.6 SUMMARY

This exercise produced several results. The patterns of QOC use confirmed those observed in the other studies. The evidence suggested that more work needs to be done on formulating heuristics to guide DSA, possible directions for which are considered later [§10.5.3]. The process model succeeded in delaying decision-making until the design space had been laid out. It also served as an analytic aid, highlighting the fact that the designers engaged in very little restructuring of the design space, in the sense that the DSA approach proposes. Possible reasons for this, and other difficulties with using QOC have been related to the notation, and the structure of the problem. It is also suggested that the difficulties experienced with QOC may be best considered in terms of how groups use representations, a line of thought picked up in general discussion [§10.4]. With respect to developing the process model, prospects for systematising design space analysis are considered in the general discussion [§10.5].

Chapter 9

Case Study 3: Organisational Issues for QOC

9.1 Introduction and method

This case study set out to observe the use of QOC in a software project in an industrial context. The team underwent a half-day training workshop introducing them to design rationale as a concept, and QOC in particular. It was hoped that the team would then attempt to use QOC in whatever way they felt useful over the following months. As described below, however, it was decided that explicit use of QOC was not suitable for much of the work in which they were engaged, although towards the end of the period of study, Options and Criteria were dominant concepts.

Most of the data for this case study is therefore drawn from a subsequent interview with the project manager, who initiated the involvement of DR in the project. In this interview, he explained why QOC was not used on the project, and the prospects for QOC in relation to their work. This case study is best treated, therefore, as an example of the organisational problems involved in successfully introducing DR to commercial software design practice.

9.1.1 The design project

The design project (called ‘Impact’) at the time of study involved a team of six people, working within the information technology division of Nestlé Rowntree, York, itself part of a large multinational foods and confectionary organisation (Nestlé). The Impact project is a pilot project to explore the requirements for eventually implementing a system within Nestlé worldwide for production planning and scheduling. Although the York team is relatively small, its work involves communicating across an international divide, as described below.

At the start of the case study, the project was beginning to define the functional requirements for the two classes of planning which the eventual system should support. These are *high level planning*, which is the process of converting sales plans through to production requirements at factories, and *low level planning* which involves scheduling resources onto machines on a shift-by-shift basis at factories.

During the period of study, the team elicited initial requirements from site visits to UK factories, and were in the process of evaluating commercially available packages in relation to these requirements. Specifically, after a week-long series of meetings with representatives of Nestlé in Europe and the US, a set of criteria was agreed (Appendix 15) for deciding whether to use an *existing* low level planning application, or build an in-house version (it was already clear that support for high-level decisions needed to be specially designed). The participants in these meetings were now being asked to rank the *importance* of the criteria which had been generated (as *essential, desirable,* or *not essential*), so that the York group could make a decision.

The period from the initial QOC training workshop, to the final interview reviewing QOC in relation to Impact, was six months.

9.1.2 TRAINING IN QOC

The Impact team requested a DR workshop shortly before beginning site visits to UK factories to gather requirements. The project manager was the prime mover behind this, having been a subject in Study 3, and seen QOC's potential usefulness. Although at this stage the introduction of QOC did not involve any new technology, he was in some respects similar to the 'champion' for new ideas within the organisation, whom Bouldin (1989) has identified as crucial to successful technology transfer (demonstrated in the DR industrial case study reported by Yakemovic and Conklin, 1990).

The team underwent a shortened version of the Study 3 tutorial [§5.1.5], plus a longer introduction to DR, and more examples of QOC. Details of the training are shown below:

- training exercise 1 (QOC analysis of small extracts from design discussions) performed individually, after which solutions were discussed together;
- training exercise 3 (scripted video) which was viewed once straight through, rather than in two halves, and then represented as QOC, performed individually;
- the ATM design exercise, tackled as a group.

The training lasted 3.5 hours in total.[1] At the end of the workshop the team expressed interest in continuing to use QOC.

9.2 Results and discussion

The results of this case study do not take the form which was initially expected, in so far as QOC was not used by the group in any explicit form, although as described below Criteria did figure largely in their work. Instead, this chapter describes how *knowledge* of QOC's constructs may have directed subsequent thinking, and identifies the factors within the project which determined that QOC was not used explicitly to record decisions or to structure meetings.

1 The training procedure developed for Study 3, and used in this study, was substantially revised for Case Study 2.

9.2.1 BENEFITS WHICH QOC MAY HAVE PROVIDED

The project manager was very positive about QOC overall, particularly with respect to its potential for improving meetings. He was asked what role he saw QOC concepts playing, if any, in their work:

> I think the main benefit that we got is that we are rationalising fairly clearly this whole business of Criteria and Options. So whilst we haven't produced lots of diagrams in the approved form, and we haven't really consciously set out to use the methodology, I think we have benefitted quite considerably from the day's session you did with us.
>
> *We weren't sure how successful that had been.*
>
> It's impossible to know how successful these things are until 6 weeks later when everyone's forgotten the day... I think that the fact that we were able to focus [the meeting] very clearly on 'What are the criteria we are trying to establish, and how important are they?', was probably a good tool on getting that meeting... essentially everyone went away feeling that it had succeeded very well, and that we had actually come out of it with a mutual understanding and an agreed consensus.... I think that at the end of it we felt it had gone a lot better than expected.

He went on to say that he could see them using QOC in a more complete form than they had thus far:

> ... I think we're beginning to find our own way into using the approach, and I would guess that having gone through that to some extent, we'll actually start to realise that some of the techniques in terms of how you draw and record things, start to make sense to us.

He also commented that in the past, lack of structure in meetings had been a problem:

> [QOC is] a good way of focussing a group workshop, because, it's difficult to say, but certainly during phase 1 of the Impact project there were a number of workshop sessions ... where the idea was that the team went along to senior management and said this is what we're thinking, and the management said 'Gee that's wonderful' or 'what a load of rubbish', and it was generally accepted that these were a walking disaster area, and that the way in which that was done didn't have any meaning to anybody.

The above comments would appear to lend support to QOC as a notation with validity as a design representation. However, as noted in the other studies, there is a primary difficulty in reporting examples or points of view such as these, namely, it is impossible to know to what extent use of QOC contributed to the views expressed.

In this case study, there are three variables which confound claims made for QOC's effectiveness. Firstly, given a meeting oriented towards generating *criteria* for assessing *alternative* packages, it would be strange if QOC was not perceived as being a potentially useful representation. The second point follows from the first: the meeting was probably better focussed than the previous ones mentioned above, making a positive outcome more likely. The manager commented to this effect:

> So in a sense, we've done something with this session which was slightly more focussed, because we had a better view of what we were trying to achieve, but undoubtedly it was a more successful week than they had before. Now whether that was due to design rationale, or because we organised it better, or just because it was a different group of people, I have no idea. You can never ever prove these things.

Thirdly, when asked, the manager could not tell if they would have structured the requirements in another way, had they never heard of DR (i.e. not as a list of criteria ranked for each package); one could argue that a matrix was the obvious way to play alternatives off against criteria, so what extra did QOC have to offer? The first two points are open questions, which can only be answered through comparative studies. Conducting such experiments is fraught with complications, however, as designers and domain need to be controlled, and reliable measures devised of the quality of design, and the utility of DR.

The answer to the third point – what extra would QOC have to offer against an evaluation matrix? – is probably 'none' in this case. The power of design space analysis comes in identifying key dimensions of a design, and elaborating the space of possibilities around each one. In this case, there only was one Question, namely *Which planning support application should we have?* This could not be meaningfully refined into separate Questions, so it is not clear that QOC had any extra analytical power to offer at this stage in the project.

The manager was asked if there were other issues which needed resolving:

> There are knock-on decisions hanging off it – which computer do we buy? Is it an RS/6000, an HP workstation, a PC running Xenix... how do we network it?, and all sorts of decisions like that, but most of those I hope will fall out once we actually get the technical options clearly established.

9.2.2 REPRESENTATIONAL ISSUES

If QOC, or DR in general was used, what would the most appropriate form be? When asked about different representations, the manager replied that the most likely form in which they would summarise their assessments was as a bulleted list, showing how each alternative scored against the criteria. He went on to observe that one of the problems with a graphical entity-relationship notation is that it has to be taught to outsiders. These overheads were the prime reason why it was not eventually used to structure and/or document the criterion generating meetings:

> ... The problem with summarising it as the specific DR diagrams, is one then has to explain the diagrams to the people you're sending them to, and that's part of the reason we stayed away from using that explicitly in our week's get-together, because we felt there were enough unknowns without throwing in a new approach to recording the meeting. We stayed as simple as we could. Even boxes and dotted lines you still have to explain to people what it is. When what we're generating is getting circulated very widely in the Nestlé world, anything that requires prior explanation is a slight problem.

One solution to this problem would be to use matrices, a widely recognised representation, instead of nodes and links. Thus each Question in a QOC analysis, could take the form of an Option-Criterion matrix (as in MacLean et al, 1989 and Lee, 1990). When this was put to the manager, he agreed that this would be an improvement.

The problem with representational form was encountered in the only other study of DR in an industrial setting to date. Conklin and Yakemovic (1991) report serious difficulties in some cases with using the indented-text IBIS (itIBIS) format outside the core group being studied, to the extent that itIBIS rationale had to be 'translated' into normal text summarising Positions and Arguments for each Issue, before minutes of meetings could be circulated to external groups. Although directed graphs are relatively commonplace within the software development community, it is likely that a hierarchical text outline is still more familiar. Given the problems that even this caused, one might conclude that the concern expressed in this case study about QOC graphs highlights a genuine representational problem which will be encountered in future work.

9.2.3 QOC FOR COMMUNICATION

The manager was asked if he could see potential uses of QOC within the project at a later stage. He foresaw QOC representations being used as a medium of communication to help maintain understanding between parties within the main Impact project:

> This project is quite small, and within ourselves we don't have great communication problems... When you get something bigger ... 20-40 people spread across 3 or 4 different countries doing the sort of full scale Impact implementation as a 2 or 3 year project, then I think some form of design rationale would be extremely helpful in actually communicating between the different groups.
>
> Because the kind of thing that tends to happen is that project teams of that size rapidly degenerate into a series of subteams each going in their own little direction, and some way in which the different bits of that process could let the other bits know why they were doing things the way they were doing them, and that it wasn't because they were just stupid, or that the right salesman had sent them a bottle of whiskey at Christmas – that there was actually a good solid reason, and that they had actually stopped and thought about it...
>
> What DR does is that it at least assures the rest of your colleagues that you've stopped and thought about it, and I think that's quite helpful in this kind of environment because people won't just accept an arbitrary decision.

As described in the last section, however, QOC would have to be in a form acceptable to different parties, and an understanding exist between project members of the role which DR was to play in project-management and communication.

9.2.4 WHAT RATIONALE WAS THERE IN CURRENT PRACTICE?

One of the frequently made claims is that rationale for decisions gets lost over time and as personnel change. It was interesting and relevant to consider therefore what – if any – forms of DR existed, however informally, in the Impact project's current design practice.[2] To what extent did the project's current practice enable rationale to be retrieved, apart from their own memories?

---

2 This is an issue on which empirical data would be most valuable. A stronger case can be made for structuring DR if it can be shown empirically that it is already recorded in a number of ways informally.

The Impact team had recently passed a project milestone, which was completion and documentation of the site visits conducted to elicit requirements. This documentation showed which issues were raised at which sites, and the manager pointed to this as an instance whereby certain classes of queries about decisions might be justified. He acknowledged however that *between milestones*, "a lot of decisions just happen" and there was no formal system for either documenting decisions, or having to rationalise them to others (although the latter presumably happens in much collaborative work).

A question which has arisen in previous studies is the level of detail which a DR should provide, and which is reasonable to expect designers to record. When asked about this, the manager commented:

> ... whether a particular field is white on blue or red on green, and this sort of stuff, is very difficult to know if you'd want to do it [DR] at that level, or whether you take a series of fairly arbitrary decisions and just live with them – at some level I think you just do that.

This response is, in this author's experience, typical of many designers' initial reactions to the concept of explicit, structured DR. It is often understood to mean "explain every decision you make" and, not surprisingly, this elicits a negative reaction. In some cases, issue-based reasoning is an inappropriate, or incompatible mode of working (as described in Case Studies 1 and 2); in other cases, it is not possible to justify everything by appealing to established theory (the case in much user-interface design); and in still other cases, the decisions are so mundane, being part of the established knowledge and training of the designer, that there is no reason why an experienced designer would query it.

An added complication is that what is an 'obvious' or inconsequential decision to one member of the design team, may be to another important enough to merit explicit justification. The above example of the use of screen colours is an example where there are well-founded human factors guidelines on the use of colour, which are often unknown, or dismissed as common sense (although what is 'common sense' to some designers is evidently not always an accurate conception of users – Hammond et al, 1983). Perhaps the only conclusion which can be drawn is that when there is no domain expert present (human factors, programmer, hardware, etc.), reasoning in that domain often remains relatively general, and is either not recorded explicitly as DR ("it's common sense"), or is reflected in the use of general arguments appealing to high level Criteria ("ease of use", "ease of implementation").

9.2.5 Sources of 'non-rational' reasoning

Within idealised models of design, decisions are made on the basis of 'rational' design reasoning, as defined by the formalised knowledge in that area (i.e. what students are taught), as well as by the expertise which accumulates with experience. In all the examples of DR presented in the current research literature, the reasoning shown always appeals to rational criteria and constraints. However, other researchers have been emphasising the *organisational context* within which design takes place and the associated influences which often intrude on decision making, often much to the frustration of designers. Grudin (1991), for instance, describes some of the organisational forces which shape poor user-interfaces.

The present case study was the only one which afforded insight into organisational issues with which DR would have to wrestle if it was used as part of everyday design. Although QOC was not used explicitly, weighting criteria – a common task in using QOC – succeeded in bringing organisational politics to the surface:

> ... so in your terms I think we're at the stage of having an agreed set of Criteria, and we're now trying to determine the relative importance, because that's where it gets very political, and that's where the sound of axes being ground starts to become very clear. It's very easy to list all the Criteria. You start to pick up where people have a political bias when you find the weightings they attach to Criteria. Because normally what people do is weight heavily the Criteria that support their preferred outcome.

The manager went on to describe instances of factors which frequently come into play in organisations:

- ❑ sites already using one of the planning-support applications being evaluated do not wish to change; having to switch to another would involve loss of personal prestige by the heads of the respective technical divisions;
- ❑ related to this, if an existing application is chosen (as opposed to designing one in-house), it is likely to be trialed at a site already using it, boosting the prestige of that site.

If QOC, or a policy-making notation such as IBIS was being used to debate this problem, how would it manage the weightings on criteria which would result from stakeholders' interests? There are at least two possibilities:

- ❑ As claimed by proponents of DR, using such a representation forces parties to be clear in their reasoning, and *exposes* "axe grinding, hand waving and clever rhetoric" (Conklin and Begeman, 1989).
- ❑ Alternately, there is the danger that the representation in fact *hides* the irrationality of a decision, through its deceptively 'rigorous' visual form which suggests that all the trade-offs have been carefully weighed; this impression is even stronger if argumentation has been weighed on a quantitative basis, as with SIBYL (Lee, 1991).

A counter-argument to the second possibility, is that the *weightings* given to goals (or at least the important ones) should themselves be justified. It is only at *this* level of argumentation that political axe grinding would become apparent; the 'surface' DR would most likely appeal to valid Criteria (at least superficially), and might at most indicate the different weightings assigned to Criteria and Assessments. It is the *justification for using* those Criteria and Assessments, and the *priority* assigned to them which needs to be made explicit.

9.2.6 THE NECESSITY OF ADEQUATE TRAINING IN USING DR

In the light of events following the workshop, it is possible to see how the training could be improved. An important factor in the acceptance of QOC is that each member of the team is able to use it without undue difficulty, for instance, to take meeting notes, as in Yakemovic and Conklin's (1990) study. Unfortunately, the resources in terms of available time on the part of the Impact team were too limited to enable this kind of training; the scope of the half-day workshop was to introduce the basic concepts, and offer some preliminary experience in using a DR notation.

Commitment by a project to learning and use of DR is essential before its merits and limitations can be fairly judged. The difficulty is that a real project running to deadlines cannot be expected to 'buy into' a new way of working unless they know what they are letting themselves in for. Thus, a cycle develops whereby initial, limited commitment use of QOC after brief training is not fruitful, prejudicing opinion against further investment in the approach. The parties involved must understand the necessary level of training to give the approach the chance to be used effectively.

9.2.7 DEVELOPING A DR HANDBOOK

As part of the DR workshop training materials, a short, informally written manual was prepared [Appendix 14], which gave a brief overview of the rationale behind DR, summarised the basic features of the QOC notation, illustrated QOC applied to different kind of problem, and emphasised that generating good QOC structures often involved working through rough versions (a theme further emphasised in the revised tutorial in Case Study 2 – §8.1).

As the prospects for systematising design space analysis with QOC are considered [§10.5], it is anticipated that more heuristics for performing QOC analyses will emerge [§10.5.3], and the benefits of certain representational forms for QOC should become apparent [§10.3].

9.2.8 SUMMARY AND CONCLUSIONS

This case study raised a number of factors relating to the use of DR in organisational contexts which are not considered elsewhere in the thesis due to the methodology adopted, and which have yet to receive much attention in DR research more generally.

In the software project studied here, issues were raised with respect to QOC's representational form, the granularity of decision which designers are expected to record as DR, and sources of non-rational Criteria due to organisational politics. There was also evidence, however, that the constructs of QOC were relevant to the problems being tackled, and may have helped to focus meetings. Its potential role was particularly emphasised as a medium of communication within projects where member parties are diverse, both geographically and in expertise.

If DR can be studied in use within industrial settings on a realistic scale, the organisational implications should become clearer. Conklin and Yakemovic (1991) highlight a number of other questions to which 'organisational DR' research could address itself. Couched in terms of QOC, these include:

- how does the organisation reward, and protect employees who honestly document poor Options?
- what is there to stop QOC being used as a means of rewarding employees on the basis of the number of Options and Criteria suggested or Questions resolved?
- what prevents QOC from becoming a tool of organisational politics, as a medium of persuasion? (– a pertinent question given the political undercurrents described earlier).

Of course, problems such as these confront any technology which mediates and records the design process explicitly, error strewn as it is. Written minutes, audio/video recordings of meetings, electronic conferences all embody design rationale in different forms, and each could potentially be used against individuals, rather than as elements of a project's or organisation's collective memory.

Chapter 10

General Discussion: Emergent Themes

This chapter summarises and draws conclusions on several fronts, but also opens up fresh avenues of thought as it explores some novel perspectives on DR use. It draws together from the different studies observations and conclusions on the benefits of QOC, and requirements for DR tools. It explores two issues which have arisen numerous times in earlier discussion, and which merit further investigation, namely DSA methodology, and the notion of 'good' representational form in QOC. It also considers two approaches to conceptualising DR use, drawing on existing frameworks in HCI research. The first recruits a range of 'cognitive dimensions' for describing information structures and in so doing, touches on several of the main thesis themes; the second approach frames DR authoring as a group design activity in which shared representations take on new roles, and considers the implications for some of the empirical data collected. Lastly, the discussion attempts to set in perspective the recurring tension between historical, narrative DR, and retrospective, rationalised DR.

Before surveying these broader themes, however, let us start by comparing the results from studies of DR use reviewed earlier [§2.1.9], with what is now known about QOC use in a range of contexts.

10.1 REVISITING EXISTING STUDIES OF DESIGN RATIONALE

It will be recalled that Lewis et al (1991) argue that design is inherently problem-centred, and that it is unnatural to abstract away from ideas in order to use argumentative DR [§2.1.8.1]. This leads them to represent DR as a series of problems and alternatives, expressed informally. The difficulty with this critique is that in seeking to minimise DR's representational overheads, their final representation leaves rationale *implicit* and *concrete*. Implicit, in that a lot of work must be done (especially by an outsider) to reconstruct the reasoning for a decision, and the possibility for computational support over the DR is limited; concrete, to the extent that no attempt is made to set the issues discussed in a wider context (generalisability), limiting its reusability.

It is intriguing however to find that incidents presented as examples of the problem-centred approach in action appear to be very similar to patterns of activity observed in QOC use. For instance, *"micro problem spawned to evaluate design alternatives"* is one of the ways in which designers generate new Criteria. The only difference is that the key property of the problem would be expressed explicitly as a Criterion.

Other problem-centred ‘moves’ through the design space also appear to be very close to QOC moves, e.g.

- *reframing a raw problem in response to design difficulties* (= reformulating a Question)
- *micro problem derived from raw problem, design alternatives spawned by micro problem* (= consequent Question derived from more general Question, Option generated from consequent Question).

This is not to say that every transition between problems need be represented explicitly using QOC or another DR notation, but those which help to structure the design space could be.

Although the problem-centred approach is weak in this respect as an approach to DR, it offers useful insights into the natural process with which DR must be integrated. If design really does proceed in a problem-centred fashion, we need to consider how DR should be recorded during design. Thus, instead of rejecting all attempts to represent a more abstract view of the reasoning, the challenge is to integrate DR representation with a problem-centred mode of working. Far from being an alternative to explicit DR, Lewis et al’s characterisation of their experiences describes *the process* through which designers’ understanding develops—*the process* by which explicit rationale is typically constructed. The two can co-exist, because there is no claim that design proceeds in cycles of <ask a Question, generate Options, generate Criteria, make Assessments, ask another Question>. Indeed, the results of the studies reported here demonstrate conclusively that typically this is *not* how QOC is used.

Lewis et al suggest that *Problems* could be introduced (i) as a new kind of DR Issue which simply asks *how to solve problem X?*or (ii) as IBIS Arguments (or QOC Criteria) to evaluate alternatives. Process-oriented DR already uses Issues and Arguments in these ways if they arise during a meeting in that form. DSA uses problem-centred Questions, but with the proviso that they may be reformulated subsequently to ask more incisive Questions in order to make important distinctions explicit. One problem which occurred with QOC (and presumably with other DR notations) is that it was sometimes difficult to understand the Assessment between an Option and Criterion because their names were so cryptic. Whilst QOC’s bridging Criteria are intended to contextualise more general principles/goals, it may be that simply summarising a problem more informally is the least disruptive, and most informative solution. This convenience comes, of course, at the cost of being unable to integrate such Criteria into a Criterion tree.

The Yakemovic and Conklin (1990) field study raised several issues of importance, such as DR training, the expressiveness of notations, and methods of assessing benefits. Some of these are addressed in the QOC studies, and others (like organisational issues) require studies of a different scope. Yakemovic and Conklin's finding that itIBIS→gIBIS conversion resulted in error detection is a pointer to potential benefits when DR is used seriously. However, since *any* revision of decisions might have led to the same discoveries, the extent to which the detections were due specifically to use of DR is open to dispute. The difficulty, of course, is that controlled studies are extremely difficult to set up in realistic design contexts.

The Fischer et al (1991) reports of PHI use are extremely interesting. The primary assertion is that, "argumentation arises out of construction, and is often tested by construction." The QOC studies support this conclusion: QOC flowed most naturally when designers 'immersed' themselves in the design, working much as they would normally for periods. They might then pause to record a new Option or acknowledge an important issue which had been uncovered as a new Question, which would then drive further discussion. However, the definition of 'construction' needs to be broadened beyond discussion and representation of concrete artifacts such as sketches or prototypes, to include scenarios and deliberation over issues not directly associated with construction of the final solution.

Lastly, the Rein and Ellis (1990) report of groups' difficulties in using rIBIS serves to confirm firstly, that without training even simple DR notations can be very unnatural to use, and secondly, that great care needs to be taken in designing the interfaces to DR tools. The recommendations for DR environments from the QOC studies are summarised in §10.7.

10.2 Observed benefits from using QOC

Much of the discussion of the data has addressed authoring difficulties which designers encountered. This is largely because the studies were investigating initial learning and use of QOC, and the problems clearly did outweigh the benefits. There were however many pointers to the way in which DR notations can be expected to support design reasoning. Listed below are beneficial properties of QOC observed in the studies:

- The QOC acted as a concrete record of what had been discussed; designers would pause to review what they had discussed and regroup thoughts.
- The QOC structure appeared to encourage completeness in evaluating Options; since Options and Criteria are permanently displayed, often as soon as a new Option was suggested, it was immediately evaluated against the 'battery' of existing Criteria; similarly, as soon as a new Criterion became apparent, Assessment links to all existing Options in that Question could be added.
- As with any (particularly visual) formalism, the spatial arrangement of the structure makes certain information salient:
  - Missing Assessment links were easily spotted by checking the Criteria;

  - 'Dangling' links were sketched from a parent Question to act as placeholders for sub-issues which needed to be addressed.
- Both design teams in Case Study 1 delayed decision making until the space had been laid out and commented on how useful it was to have a clear representation of the issues, as they went through boxing Options.[1]
- The process of constructing QOC encouraged the Case Study 1 designer to clarify the advantages of a favoured Option; he also found it helpful in 'tying down' many ideas which had been up to that point disorganised and not fully articulated.
- An informal observation is that compared to Study 3, which used Issues instead of Questions, designers in the other studies seemed to take more time in focussing on what the real issue was—having to formulate *Questions* helps to preempt simply jotting down a couple of words as an issue.

In concluding their report of the itIBIS field study, Yakemovic and Conklin (1990) ask if there is any "magic" in IBIS as a notation, which made it useful as a rhetorical model. The above findings are evidence that representing arguments as explicit structures yielded benefits both in the process of reification, and afterwards in seeking to reconstruct reasoning. This is in addition to the many small incidents embedded in the renaming and restructuring involved in honing more precise structures. There is however not enough evidence at present to show whether the kinds of insights QOC offered into the designs provide sufficient payoff to merit its use; large scale designs and QOCs are needed to see if the improvements which designers make to their analyses are mostly trivial, or offer deeper insights. It should also be emphasised that such benefits are most likely to arise with designers who are *comfortable* with QOC (as opposed to the designers studied here), using it for their *own work* (as opposed to experimental tasks). In this context, a QOC comes to be treated as a familiar, external memory, much richer in significance than the skeletal representations necessarily generated within the constraints of laboratory study.

Lastly, the studies reported here were not designed to demonstrate that better designs result from using QOC, although this is clearly one of the goals towards which the DSA work is directed. Design creativity is hard to define, quite apart from judging it, although some attempts have been made to formalise the concept (Thomas et al, 1977). It may be that better design from using DR will only result via less direct means. For instance, rather than being able to prove that using a particular notation yields particular kinds of insights *during creation*, the payoffs may come *in reuse*, such as the time and effort saved in retrieving decisions or identifying dependencies. This is however still very much an open issue.

---

1 This is particularly interesting, as the tutorial in this study emphasised strongly that decisions should be delayed until the end. Study 3 designers did not benefit from this aspect of QOC to such an extent.

10.3 A REPRESENTATIONAL 'STYLEGUIDE' FOR QOC?

In the course of conducting these studies, it became clear that there were certain 'groundrules' about the way in which QOC structures should be represented. As one designer stated, "we need rules for DR." QOC which conforms to these rules (i) encourages clearer thinking on the part of the designers constructing it, and (ii) is more intelligible to others (and hence reusable). MacLean et al (1991) have suggested some properties for Criteria, and it is in a similar mould that properties of other QOC constructs are suggested here. Table 10.1 lists some properties which have repeatedly suggested themselves as characteristics of well-crafted QOC analysis, both from subjects' comments, analysis of their QOC representations, and personal use of QOC.

Questions

Q1. There are no 'Yes/No' Questions

These effectively ask *Should we do Option X?* and do not encourage the designer to enumerate the possibilities around that Option.

Q2. Each Question addresses only one important issue

This allows Options to be assessed with respect to that issue. If there is more than one embedded issue, different Options respond to different requirements, and the phenomenon of 'Criterion bunching' occurs, whereby several Criteria are relevant to and assess only one Option, whilst the rest assess another. Another way to express this is that *For a given Question all Options should be assessed by all Criteria.*

Q3. Questions do not ask how to achieve Criteria

A Question such as *How do we reduce errors?* asks how to satisfy a requirement or goal, which are best represented as Criteria. The Options which such a Question spawns are often other subCriteria, not design decisions (see O3).

Q4. Questions do not summarise the Options

Questions should not ask *Should we do X or Y?*, with Options *X* and *Y*. The Question should pick out the common function which X and Y serve.

Q5. Generic Questions are used whenever possible to capture a potentially generalisable principle across the design space

If several instances of a more general Question have arisen, it is useful to make that commonality explicit by formulating the generic Question. The general question need only be asked once, after which specific instances are answered by default.

Options

O1. At least two Options are considered for a given Question

This is a basic characteristic of the design space analysis approach—designers are encouraged to construct and explicitly represent the space in which their design is located.

O2. Each Option has positive and negative Assessments

In order to minimise confirmation bias in decision-making (the tendency to seek confirmatory as opposed to contradictory evidence to one's preferences), the downsides to Options should be recorded.

O3. Options are potential design decisions

Options represent the 'moves' through the design space which map out the path taken in a particular design. The goal of a well-structured design space is to capture a range of potential designs in a given domain; Options must therefore be specific design features, or closely related to features. Questions which generate Options which are really Criteria, are about strategy (e.g. *Q: How to speed up task? O: minimise learning costs*).

O4. There is one design alternative per Option

Options which really embed two Options create representational difficulties in evaluation. Criteria assessing an unfocussed Option may be addressing different facets of that Option. Unfocussed Options also defeat the purpose of using a notation which allows elements to be added incrementally, and in a form which supports computational support for retrieval and manipulation.

O5. Options are expressed at a consistent level of detail within a given Question, or: Options are generally distinctive alternatives, not minor variations on one Option

The initial emphasis should be on representing the 'boundaries' of the local space (defined by the Question) as contrasting Options. Once a decision has been made, a consequent Question might then focus in on the microspace around a particular Option—those Options should not however be enumerated before then, in order to maintain a consistent level of detail.

Criteria

C1. Criteria are positively expressed

This maintains consistency in the semantics of Assessments (a *supports* link always represents a 'plus' for an Option).

C2. Criteria are not duplicated within a Question

Often designers find that they have effectively re-expressed a Criterion as another Criterion (a clue being that the two make identical Assessments for each Option). Unless it is valuable to make two subtly different manifestations of the same Criterion explicit, it is more elegant to use only one.

C3. Criteria only embody one goal

As with Questions and Options, Criteria should be focussed. If a Criterion integrates more than one Criterion (= goal), then it is ambiguous–which embedded goal are Options being assessed against?

Assessments
A1. Weak Assessments are neutralised with better Options
The goal in design is to make decisions with as few weaknesses as possible. If a decision (or preferred Option) has a significant negative Assessment, the designer will seek Options which satisfy the objecting Criterion as well as maintaining the other positive Assessments. In QOC this may be through asking consequent Questions, or overcoming the problem elsewhere in the design space.

Table 10.1: Some candidate principles for a 'styleguide' to QOC representational form.

As they are purely syntax-based, some of the above are amenable to automated verification, via *schemas* which define the notation (which entities can be linked to which, via which relations). They are particularly useful in knowledge structuring hypertext systems such as GERM (Bruns, 1988) and Aquanet (Marshall et al, 1991). Schema could check for many of the structural patterns in Table 10.1 such as single Options, missing Assessments, decisions with negative Assessments, or Yes/No Questions.[2] However, other instances of poor representational form require deeper semantic or content analyses of the relationships between entities, e.g. Questions about Criteria, lack of focus in an entity (Q/O/C), or the existence of generic Questions. Indeed, some of the latter are often extremely difficult for experts to decide, quite apart from automating the process.

One of the emphases of this work has been on the need to support the emergence of 'rational' structures from the unstructured and sometimes vague and chaotic mass of ideas which are the stuff of early design deliberation; clearly, designers will not express ideas in such a fashion as to satisfy the above 'styleguide' immediately. The purpose of moving in this direction is to make available representational 'scaffolding' which will render DR manageable and reusable for others, but this inevitably places an extra burden on the designer. The tightrope is being walked between on the one hand regimenting QOC structures so tightly that the flexibility and creativity germane to creative design is lost (a common criticism of formalisation), whilst on the other hand maximising the potential benefit of reifying rationale for both author and subsequent user. The above list is presented as an initial proposal, open to addition and deletion as experience and empirical evidence direct. It remains an open question as to when representational constraints (for these are what a styleguide imposes) provide helpful compositional structure, and when they become a creative straitjacket.

10.4 QOC AUTHORING AS A SHARED WORKSPACE ACTIVITY

This section outlines how one can better understand the way in which DR representations are used in group design by drawing on a conceptual framework developed by Tang

2 Note that these were also used as measures in Study 3's quantitative analysis of QOCs [§6.1.2].

(1989). Whilst many of the results reported thus far add to our understanding of the "nuts and bolts" of constructing DR with QOC, on their own they offer only a partial picture of QOC use. In order to more deeply understand QOC's effect on *group dynamics*, it is necessary to adopt a different perspective on QOC authoring which focusses on the interaction between designers, the representations they are using, and their tasks. The vocabulary of discussion moves now from low-level tasks such as node naming, classification, weighting, and linking, to a different set of concerns addressing role distribution, group activity during periods of QOC representation, and the importance of QOC structures as representations owned, and collaboratively built by the group. The analysis is developed with respect to differences which were reported between the two design teams in Case Study 2; however, the conclusions are applicable generally to the ways in which DR representations can be used.

The line of argument to be pursued is as follows: as it stands, the five-phase QOC process model explicitly proposes the refinement of rough, working QOC representations as a means of directing design discussion towards exploring the design space; Tang's analysis of shared workspace activity enriches this by introducing the distinction between the expressive and documentary roles of representations; designers' experiences that using QOC distracts from 'real' design stem from their use of QOC simply as a passive medium for documenting decisions, as opposed to a working representation used to collaboratively develop and express ideas.

Tang (1989) proposes a framework (Figure 10.1) for describing the shared workspace activity of small groups engaging in conceptual design.[3] Based on studies of how design teams used whiteboards and paper, he develops an analysis of their activities in terms of *actions* (textual listing, drawing, and gesturing), and the *functions* which these serve (storing information, expressing ideas, or mediating interaction[4]). This characterisation of group design has served as the conceptual basis for several prototype shared drawing tools (Tang and Minneman, 1990; 1991; Greenberg and Bohnet, 1991, Lu and Mantei, 1991).

3 A briefer description can be found in Tang (1991).

4 The role of the workspace in *mediating* interaction (e.g. turn-taking) is made explicit in the framework, but is not of specific relevance in the context of this analysis. One would expect a similar role to be played by any collaboratively built representation, including DR.

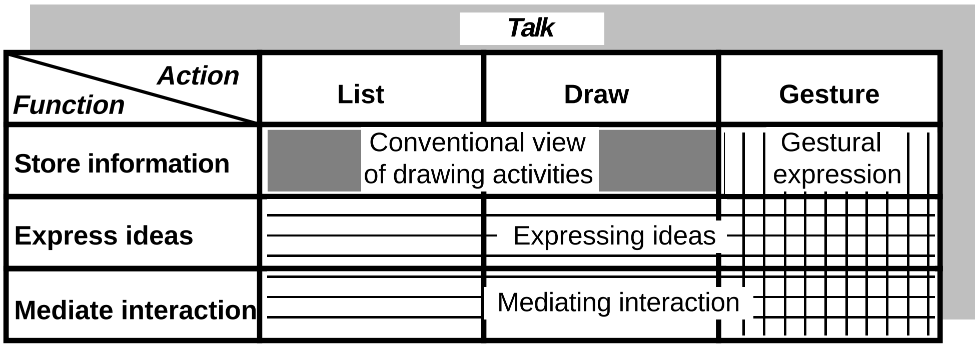


Figure 10.1: A framework for describing the workspace activity of design teams (from Tang, 1989).

The framework distinguishes between recording notes and sketches for *information storage*, typically after explicit agreement, as opposed to the *development and expression of ideas*. The distinction is between documenting 'complete' ideas for later recall, and expressing 'incomplete' ideas to enable others to react to and build on them. Clearly, ideas recorded for later recall may at any point become a vehicle for developing further ideas (and Tang presents instances of interaction which bridge categories). Goel and Pirolli (1989) also describe how representations (e.g. sketches) were used to develop ideas, as opposed to simply recording them:

> Within a single symbol system, he [the designer] constructs multiple representations of the artifact. In both cases, we want to note that these external representations are not for communicating something after the fact. They serve an indispensable role in the generation, evaluation, and decision-making process. Once decisions are made, symbol systems serve to record and perpetuate them. [p. 32]

A DR is evidently an artifact for storing and retrieving information; moreover, it has been hypothesised that representing DR during design actually facilitates the process. Intuitively therefore, Tang's framework suggests itself as a potentially useful way in which to characterise group use of QOC. Let us now consider its implications for QOC use in a collaborative setting.

10.4.1 Recording QOC as a documentary activity

In documenting completed, agreed upon ideas, the aim is to record the information in the most efficient, timely manner possible. However, the delay this introduces creates a problem for a group striving to maintain momentum in a meeting. Tang observed that a group can manage the delay in three possible ways:

1. the rest of the group waits for the scribe to finish recording ideas;
2. the rest of the group occupies the pause with individual work;
3. the rest of the group moves on to discuss another issue.

In Case Study 2, the FileGroup design team used the third strategy, and reported that the scribe was left out of discussion and had to catch up. In general, the second strategy did not occur in this study, as a single scribe was appointed for recording QOC, and hence controlled the representational workspace.

A difficulty with classifying representational activity as 'information storage' rather than 'expression,' is that in design, commitment is often delayed, so decisions may be changed. Thus, whilst it was common for designers to watch whilst QOC was recorded (the first strategy), this often drew further contributions, such as suggesting a new Criterion, or a better name for a node. The examples of node naming in Study 3 show that it is often an intensive group activity, and that the process of recording ideas which are initially judged to be complete often leads to further refinement. In sum, an accurate characterisation of QOC authoring would emphasise that during design meetings, any part of the DR is open to modification at any point; this makes it difficult to isolate (in Tang's terms) instances of "recording information after explicit group agreement for later recall." Clearly, however, supporting retrieval is critical once the DR is completed.

10.4.2 Using QOC as an expressive medium

Tang's category of *expressing ideas* refers to the use of lists, sketches or gestures in order to communicate ideas to the group:

> Rather than being intended for later recall, expressing ideas interactively elicits a response or reaction in the present time. (p. 72)

Tang contrasts the use of time for *documenting* as opposed to *expressing* ideas, as follows:

> [T]he goal of storing information is an *artifact* that records information for later recall. The *process* of creating that artifact is often troublesome, due to the time delay involved. However, when expressing ideas, the goal is to enlist the interaction of the group to develop ideas. Having the group experience and participate in the *process* of creating workspace artifacts is an integral part of expressing and developing ideas. (p.86)

He goes on to note that artifacts used or created to aid the expression of ideas (such as single words, doodles, emphases on sketches or words) are often meaningless without the context in which they were embedded (gestures, preceding discussion, shared background knowledge).

If we now consider QOC, we see a parallel pattern. Early QOC is not always intelligible to others. When used for expressing and developing ideas, names are often impoverished, sufficient only for the group to understand; similarly, problem decomposition may be partial or simply wrong, again, useful only to the immediate design group as they explore the problem. However, designers must be able to work with such incomplete representations in order to develop ideas. In introducing a DR notation to the already complex picture of design described by Tang (and others), it is important to recognise this by building tools which take into account the *expressive* use of notation.

Thus, the current process model of iteratively refining rough QOC representations, within Tang's framework, describes the evolution from a representation for *developing* ideas within the group, to one which *communicates* to others—a move from *expression* to *documentation*.

We can now return to the specific difficulties with QOC encountered by a number of designers, the FileGroup designers being a case in point. Their main assertions [§8.2.4] were (i) that their usual way of working already embodied much of QOC's approach to enumerating possibilities, but that being forced to explore the space was constraining; (ii) that what was different about QOC from current practice was *representing* the space explored, but the notation was inadequate; and that (iii) together, these acted as a brake on the dynamic nature of their normal design meetings.

The key to the problem lies with the way in which QOC is used. It cannot, and was not intended to replace brainstorming, or the pursuit of intuitive ideas – such spontaneity is germane to creative design, as is the momentum which a team of experienced designers generates when working well together. It is at this point that the distinction between QOC as *documentation* as against *expression* is valuable. If QOC is used only to document rationale – that is, semi-formal DR is treated simply as a form of structured minutes – it assumes a *passive* role, as it plays no part in shaping reasoning about those ideas. However, QOC can also be used *proactively,* for expressing and developing ideas, if the QOC representation plays a central role in generating the topics for brainstorming and discussion.

According to this analysis, FileGroup's comments on QOC stemmed from its use as passive documentation for decisions and rationale, rather than as a medium for expressing and developing those ideas in the first place. The absence of any enumeration of the local space around the Options (Phase 3) or of structural reformulation (Phase 4) indicates the documentary nature of the way in which QOC was used.

The extent to which DR representations will play the same roles as Tang observed for notes and sketches, rests on how DR is used. DR can *always* serve as a medium for information storage, whether it be short or long term: for a single meeting, for the duration of a project, or for outsiders to understand the design or draw on the knowledge therein for other designs. However, independent from its use for information retrieval is whether DR is used *expressively,* and this rests on the key issues of *who* creates the DR, and *when* it is created. If DR is one element (amongst other representations) of the way in which a design team conceptualises the design, it can be used as an expressive aid during design. Figuratively speaking, the QOC becomes the group's *working* memory for framing the problem space, as well as *long term* memory for storing decisions. QOC structures serve as a focus of attention for discussion – the group works together to define the Question, evaluate Options, and so forth. Used in this way, the QOC becomes a transparent medium for addressing the design and design space. This dimension to QOC will be lost if it is used simply to record decisions.

This analysis has implications for two suggestions commonly made by designers and DR researchers as to how DR might be used in meetings. One is that a designated 'scribe' should record discussion as DR, whilst the other designers proceed with the 'real design' work, free of the overhead to use DR; the second arrangement is that decisions are translated from notes into QOC by someone after meetings. In both cases, QOC plays no part in the group process which Tang identifies as critical in developing ideas. The loss of the potential benefits of QOC as a working representation are partially compensated if the structure is 'revisited' by the designers subsequently to review decisions. The QOC might at that point become a working representation on which the designers could build.

In summary, it is proposed that Tang's analysis of the use of representations in shared workspace activity has insights to offer in relation to the use of DR representations. Collaborative-tool builders have to some extent validated Tang's observations of design activity with regard to'normal' artifacts like notes and sketches (e.g. Ishii, 1990; Greenberg and Bohnet, 1991; Lu and Mantei, 1991; Tang and Minneman, 1991). Given the initial evidence reported here, the application of the framework could serve as an analytic starting-point in future studies of DR in use.

10.5 TOWARDS A METHODOLOGY FOR DESIGN SPACE ANALYSIS

> ***method*** *a special form of procedure, especially in any branch of mental activity; orderly arrangement of ideas; scheme of classification*
> (The Concise Oxford Dictionary, 7th Edition)

This section considers the next step for systematising design space analysis, following the results of Study 3 which described results from the first QOC tutorial, and Case Study 2 which reported the results of an initial assessment of a range of QOC heuristics in the context of the five-phase process model.

To what extent can a method be prescribed for using QOC? Taking the first definition given above, in most classes of design, particularly conceptual design and HCI design, it is impossible to prescribe a complete method as a formal procedure. Automated design by expert system is limited to extremely well understood domains, of restricted scope. DSA is inherently an exploratory process of discovery, in which optimal views of the space are mapped out. Consequently, 'method' in DSA is likely to develop at the macro-level in the form of general activity phases (as in the current process model), and at the micro-level as grains or fragments of knowledge encapsulated as heuristics or principles for approaching the problem at different levels of generality. Just as Halasz et al (1987) suggested that a "NoteCards strategy manual" would have helped new users learning to use NoteCards in their everyday work, one result of work on DSA methodology might be the production of a QOC strategy manual. A preliminary move in this direction was made in the provision of summary crib-sheets for designers in Case Study 2, and of a short handbook on QOC in Case Study 3 [Appendices 11 and 14].

Study 3 and Case Study 1 demonstrated clearly that on a first exposure to the philosophy of DR, and to DSA and QOC specifically, designers are generally able to learn to use QOC to record decisions. There were also signs that this level of DR use might be approaching 'saturation point' in terms of the amount of new information which could be effectively translated into practice in one day: a heuristic which introduced too drastic a change in normal design (*use extreme Options*) was not used, and Phase 4 of the process model, *restructure design space*, was effectively ignored. For these reasons, it was also concluded that global heuristics would need to be taught in a subsequent 'advanced tutorial.'

Let us consider the 'method' as it currently stands, and its development in three respects: (i) the central role of informal, intermediate QOC representations in deriving useful DR, (ii) methodological issues in evaluating a process model, and (iii) developments to the current set of heuristics.

10.5.1 Rough QOC: The importance of intermediate representations

The QOC studies reported show that creating intelligible, reusable DR takes work: how do we support designers with appropriate representations in this process? The process model of QOC authoring acknowledges the importance of *evolving views* of the design space, mediated by, and reflected in *iterations through intermediate representations of the design space* as new issues and perspectives are uncovered. *Rough QOC* is being used as a label to cover the necessary intermediate representations involved in deriving coherent DR, and has conceptual roots in two other research fields, namely knowledge elicitation and research into tool support for writing.

The emphasis on rough QOC as an approach to the authoring problem derives in part from a view of *QOC authoring as knowledge elicitation.* It has already been noted that hypertext and expert systems bear close similarities, indeed, are best viewed as lying along a dimension of 'representational formality' [§1.2.1]. It was not surprising therefore to find, quite independent of the development of the rough QOC concept, that intermediate representations have also been used to facilitate the externalisation of ideas in knowledge engineering. Gammack (1987), and Young (1988) criticise knowledge elicitation techniques which carry the implicit assumption that expert knowledge can be efficiently translated more or less direct from interview notes into executable rules. Instead, they argue for the use of intermediate representations to bridge the gulf between human and machine representation, suggesting that repertory grids, and visual representations of concept-similarity ratings were a better way to elicit different kinds of knowledge.

Of specific interest are their observations of the process of transforming knowledge representations from "human-like" to "computer-like." Three systematic changes are noted, which resonate closely with themes in this thesis:

(i) *a loss of completeness*: computers cannot possibly represent all the information which make up an expert's knowledge [DR: what information is lost in representing reasoning semi-formally? What kind of DR is of most use?]
(ii) *less semantic interpretation, and more syntactic analysis*: knowledge is parsed into structure at a finer and finer granularity [DR: the trade-off with between computability and usability has been noted numerous times – see §10.7.2]
(iii) *a loss of cognitive compatibility*: humans cannot easily comprehend or express knowledge in machine readable form [DR: this is perhaps the principle theme of the thesis – even *semi*-formal notations introduce a representational gulf].

A second thrust in knowledge engineering research is represented by the work of Bradshaw and colleagues (Boose at al, 1990; Bradshaw and Boose, 1991). They present the motivation behind several tools which make available multiple representations for knowledge engineering. Intermediate representations are viewed as bridging the gulf between *mediating* representations, the representations with which experts and knowledge engineers work to build problem-solving models, and the actual *knowledge base*. By this definition, 'mediating' and 'intermediate' representations are a refinement of Young and Gammack's more general use of the term 'intermediate.' Their DDUCKS environment comprises several tools and a wide range of representations including textual lists and outline views, matrices, process diagrams, graphs, and trees.[5] Just as with Young and Gammack, we find that many of the concerns expressed map closely to those in this research:

❑ is the formalism expressive?
❑ does the formalism aid communication within the development team?
❑ does it guide knowledge analysis in a significant way?
❑ does it make important things explicit, and hide detail until needed?
❑ does it expose natural constraints?
❑ is it complete and concise?

To summarise, effort must be invested at some point in the DR authoring process. A process incorporating rough QOC distributes the load partially on the initial recording of ideas (labelling/grouping ideas), partially over the duration of the project as understanding grows and designers reflect on the growing DSA (monitoring themes and design space

[5] DART (Design Alternatives Rationale Trade-offs) and several other tools have been developed for capturing design knowledge, and managing complex design trade-offs. This work lies towards the more formal end of systems which could support DR – the increased power trading off against the higher authoring costs.

coherence), and partially on the restructuring of the QOC retrospectively (does the representation communicate what we *now* know to be the key issues in the design space?).

10.5.2 Difficulties in assessing informal DR process models

It will be recalled that the five phases in the current process model are:

Phase 1: Organise available material
Phase 2: Structure material into rough QOC
Phase 3: Flesh out design space
Phase 4: Restructure design space
Phase 5: Make design decisions

Phases 2, 3 and 5 constitute the bare minimum needed to produce a QOC translation of some sort. Phase 1 encourages consideration of the source material before asking the first Question, and Phase 4 emphasises the frequent need to revise the analysis in the light of new insights.

In assessing the extent to which designers work according to a given process model, differences between the model and observed activity can be handled in several ways. It might be decided that the process model is:

- *inadequate*, and needs extending to accommodate the observed use of QOC, so that designers are encouraged to continue using QOC in the way that they have been;
- *adequate*, and designers should be discouraged from using QOC as they do currently;
- *adequate, but only for certain kinds of design problem solving*; other process models are needed for other kinds of, or stages in problem solving; this might produce a family of process models for different modes of working. Decisions would have to be taken as to whether a process model should be extended, or a new one developed.

To find evidence of the utility of a process model, it is not sufficient to find evidence that QOC was useful. If designers gain benefit from using QOC, the key question must be, in what way, and does the model itself (as a method to be followed) promote such ways of working, or were the designers actually working outside of its 'brief'?

One problem with assessing the current process model is that it makes very few strong predictions. Because the phases are not meant to be strictly sequential, evidence of moving from one to another has little significance, and activities cannot be analysed in terms of order. This is an inherent problem for non-linear process models, but difficult to overcome without straitjacketing designers. Perhaps the strongest support for a model is if its phases map closely to important problem-solving activities; however, such a result proves only that the model can accommodate current activity, which is much weaker than being able to prove that it contributes something additional of value. Furthermore, without many comparative studies to provide data on how the designers tackle the same or similar problems without QOC, one is left in the weaker position of citing examples where use of the method/notation *appeared* to facilitate design.

In an informal comparison of the Case Study 1 data with data from four non-QOC design sessions, Elworthy (1991) notes that the general structure of non-QOC sessions was very different, but it is not clear whether this was due to QOC or not. In the non-QOC sessions, discussion was less focussed on specific issues, and strayed from relevant ones; in addition, Criteria were not reused from one issue to another, and tended to be much vaguer than in QOC. Whilst in-depth comparative studies (of which there are currently none in the literature) are crucial for assessing the impact of DR, it was not clear in Case Study 2 to what extent the observed benefits could be attributed to QOC, far less the specific process model (due to different tasks, designers, and session durations). Elworthy suggests that one might study designers tackling the QOC design problems without QOC, and vice-versa (which ideally, would reuse the original designers in a within-subjects experimental design). However, if it is the case that QOC was responsible for the differences (which given their nature is reasonable to assume), there is still insufficient evidence to claim that the process model made a significant contribution, since clearer focus on relevant issues and Criteria, and the reuse of Criteria could be viewed as *general* properties of QOC.

Given that the existing process model phases are sufficient to cover the DSA process at a general level, one approach to extending the model is to refine the existing phases—what is actually *involved* in fleshing out, or restructuring the design space? These lower level operations are addressed to some extent by the syntax-oriented heuristics (H1-8), and by the global heuristics (not yet evaluated). However, designers reported that H1-8 (e.g. *use Options to generate Questions*) were 'too obvious,' and it was concluded that procedural principles at this level might be inappropriate as DSA heuristics. However, what might be useful are procedural rules, which have unambiguous preconditions for application, and well-specified actions at the level of structural operations. These are considered next.

10.5.3 Informal 'production rules' for QOC construction

Rather than leave guidance for designers at the level of *use Options to generate Criteria*, it may be possible to articulate representational guidelines at a lower level. The aim is to not to articulate new principles for developing QOC analyses, but to encapsulate existing knowledge at an appropriate level (where what is deemed 'appropriate' will be established through building on our experiences of QOC in use). However, the process may be reflexive: struggling to articulate DSA at the level of procedures may lead to the discovery of new principles and strategies.

Consider the following example of a 'rule' which formalises the heuristic *overcome negative Assessments with positive ones*:

IF you wish to overcome a negative Assessment to an Option

THEN ask a consequent Question specifically about how to implement that Option whilst maintaining its positive Assessments

This is illustrated below (taken from an example of this heuristic in the revised tutorial). The example illustrates one way to satisfy both *speed of transaction* and *range of services offered.* The first Option (*cash plus other services*) offers more than cash, but is slow. A highly focussed consequent Question has embedded in it the implicit requirement that multiple services are offered, and now seeks the fastest Option with multiple services. An Option is then found which is judged to satisfy both Criteria. Clearly, using this rule will not generate the solution, but may encapsulate a simple principle which draws the designer's attention to potentially relevant parts of the design space.

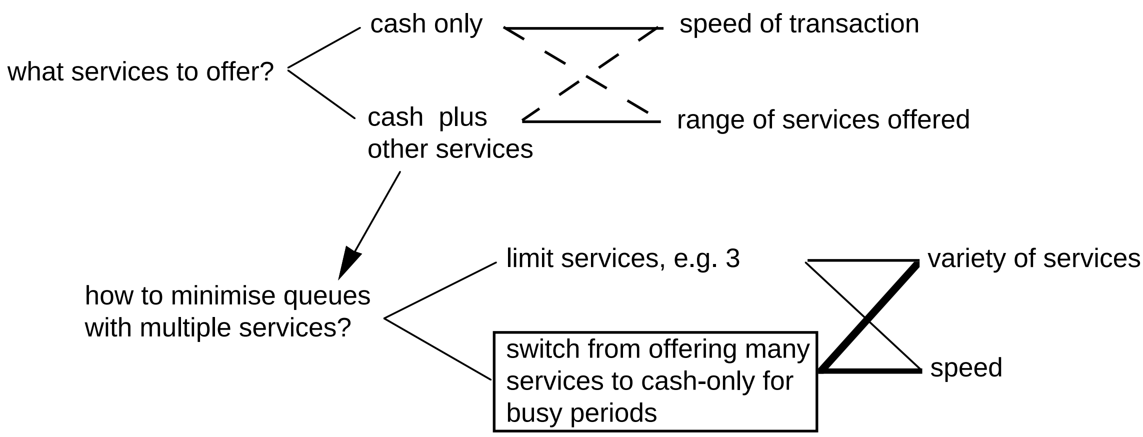


Figure 10.2: Using a representational production rule to make explicit how to apply the QOC heuristic "overcome negative Assessments."

Expressed in this way, these are functionally equivalent to production rules, though expressed informally. However, the precondition is well-specified, and the action is expressed at the level of rhetorical moves in QOC (ask a *consequent Question...*) and even provides guidelines as to the content of the Question (*...about the Option, maintaining positive Assessments*).

The above rule makes a common QOC task explicit, and provides guidelines as to how to 'implement' it structurally. Other examples are shown in Table 10.2.

IF	you wish to group two Criteria into one because their difference is trivial in the context of the Question
THEN	it is necessary to delete one set of Assessments (Assessments to each Criterion should already be identical) and the new Assessment weightings should be very similar to the original ones *(see Figure 7.4)*
IF	you wish to decompose a general Criterion into two bridging Criteria
THEN	duplicate the Assessments currently held by the general Criterion, but the weightings may be different to each bridging Criterion. *(see Figure 7.4)*
IF	the first idea to come to mind is a Criterion (i.e. a goal to meet)
THEN	it can be helpful to ask a Question about how to satisfy it; the Options to that Question will be more focussed Criteria
AND	those Criteria can be used to assess Options in subsequent Questions

Table 10.2: Examples of QOC representational production rules.

It is hypothesised that procedural rules such as these will be more easily understood by designers once illustrated, and more easily applied as they are grounded in the tasks (the conditional IF...) for which they are relevant. Next steps would be to extend the set of rules, and to evaluate the usability of knowledge encoded in this more focussed form.

Whilst heuristics and production rules provide a language for designers to conceptualise QOC tasks, we turn in the following section to a possible language for the designers *of DR systems* to describe and contrast their work.

10.6 *COGNITIVE DIMENSIONS* OF DR SYSTEMS

Green (1989, 1990) has proposed a series of *cognitive dimensions* which can be used to describe a wide range of 'information structures,' including programming environments, text and graphics applications, and databases. In this section, his analysis is extended to the domain of DR, using three notations to illustrate points.[6]

As Green (1989) explains, the proposed dimensions are not intended to be orthogonal like mass, length and time in physics. Maximising the 'score' on a given cognitive dimension inevitably trades-off against others, "usability" being a function of the total interaction. The dimensions which a notation optimises reflects the purposes (domain; tasks; users) for which it was designed. For instance, a system offering knowledge based support will invariably use a richer underlying domain model and be able to make explicit more *dependencies* (a dimension) than a simpler notation; however, this is at possible cost to its learnability.

6 Fuller discussion of cognitive dimensions and DR can be found in Shum (1991b).

A *system* is viewed as comprising a *notation,* the entities manipulated by the user, within an *environment* of some sort, the medium of interaction. The dimensions describe systems as *information structures*, so that lessons learned in one domain can be abstracted and considered as principles in others. In sum, any information system can be assessed in terms of cognitive dimensions, since the dimensions characterise structures in domain-independent terms.

Note that the aim is not to see which DR notation comes out 'top' after 'firing' various dimensions at them. Whilst a broadbrush comparison is in any case impossible to make, as they were developed to serve different purposes, it will be demonstrated that cognitive dimensions do offer insights into user-centred differences between notations. The analysis presented here was conducted to explore the extent to which cognitive dimensions could be usefully applied to DR, with the hope that tool requirements might also be clarified.

10.6.1 Four dimensions for describing DR usability

Four cognitive dimensions are now discussed in detail, with respect to DR. Each dimension is defined, and illustrated in the context of DR with evidence from the studies.

10.6.1.1 Premature commitment

Premature commitment describes the relationship between the order in which ideas occur to the user, and the order in which the system requires them to be entered. Problems arise when the system forces users to encode ideas before they are ready to. To paraphrase Green (1989), "to what extent are designers forced to adopt a *notational order* in recording DR, which is incompatible with the *generative order* of ideas in private reasoning or group discussion?" One example is gIBIS, which requires an Issue to be posted before Positions, and Positions before Arguments. This is a consequence of adopting an explicit rhetorical model for meetings, which can cause problems when ideas are unclear, resulting in 'premature segmentation' (Conklin and Begeman, 1989). NoteCards introduces a very similar way of working in this respect, imposing strict constraints on titling, and filing (i.e. linking into existing network) each new notecard.

Premature commitment addresses a property of notations which has a direct bearing on supporting the whole QOC authoring process, from initial ideas to rationalised structure: if designers are constrained to expressing reasoning as semi-formal DR notation or not at all, they will be forced to commit themselves prematurely to a form which their own conceptualisations are not ready to support. Furthermore, as the trade-off between expressiveness/computational power and usability dictates, enlarging a vocabulary creates more choices which designers must make about the relationships and roles of their ideas. The effect of a more elaborate language thus forces designers to be more precise in how they translate ideas, creating additional overheads, the net result of which is distraction from *analysis* of the *problem*, to *representation* of the *analysis*. Designers must be able to record the most tentative of ideas in the knowledge that attention need not be directed to naming and linking until they feel ready to; there must be no obligation to label ideas; it should be possible to indicate relationships through spatial clusters; and renaming must be the simplest of procedures.

In dealing with the 'wicked problems' encountered in design [§1.3.1.3], framing and reframing the problem is an important process. This translates in QOC to reformulating the Questions used to structure the design space, as observed in the studies [§6.2.4, §7.2.1.2].[7] MacLean et al (1990) reported that the designers studied asked "impoverished questions" which precluded creative solutions. Reformulating views places a strong requirement not only on being able to restructure easily (see 'viscosity' [§10.6.1.2]), but on identifying the need to restructure in the first place. In this context, the danger with an early QOC representation which conveys a deceptive 'completeness' (graphical DR appearing to constitute rigorous argumentation) is that it may prematurely impose views of the design space which carry excessive weight in shaping the final design. Thus, a poor early decision may pass unnoticed because it has backing argumentation against several alternatives, but the Question to which it responds may not address the real issue at stake. Interestingly, premature crystallisation of solutions due to the 'finished' look endowed by software tools is also a problem in typographic design (Black, 1990).

In terms of premature commitment, the paper-based environment presents few initial constraints on where or how to record ideas (Green, 1989, p.452), but has poor spatial restructuring capability. In contrast, collaborative drawing tools allow ideas to be moved, changed, and linked relatively easily, suiting them well for this mode of working. The challenge now is to build seamless environments in which one may move from these low constraint representations to ones employing richer vocabularies and offering large scale structure manipulation. One example of this (Bobrow et al, 1990) describes a demonstration tool interfacing Cnoter, a brainstorming tool, with NoteCards; it would interesting to see something similar to this tailored to the requirements of a DR authoring environment.

To summarise, in augmenting the many advantages of paper-based media, developers of DR authoring environments must avoid committing designers to encoding DR prematurely, which in idea structuring tasks often implies the use of different representations.

---

7 However, as use of the process model in Case Study 2 highlighted, during early use of QOC the overheads may distract attentional resources from engaging in much restructuring activity.

10.6.1.2 Viscosity

> For many interesting areas, a structure is not given a priori but evolves dynamically. Because little is known at the beginning, there is almost a constant need for restructuring. Despite the fact that in many cases users could think of better structures, they stick to inadequate structures, because the effort to change existing structures is too large. (Fischer, 1988)

The ability to easily *revise* entities and structures has been shown to be critical [§6.2.3, §6.2.5], and is captured by the dimension of *viscosity*. Very simply, viscosity refers to a system's resistance to change (the analogy being with a viscous substance). How much work does the system create for users making changes?

Green (1990) has refined the viscosity dimension, identifying two kinds of viscosity, and root causes: (i) *repetitiousness viscosity* is encountered when the same operation has to be repeated many times, rather than executing one global operation over every instance (the notation offers no abstraction mechanism); (ii) *knock-on viscosity* is apparent when to make one change, a ripple effect means non-goal related changes also have to be made to preserve the integrity of the structure (rather like a pile of wood, there is such high "constraint density" that nothing can be moved without disturbing the remaining structure).

In QOC, hierarchical relations between Criteria are a useful abstraction to recognise. Lee's SIBYL implements this idea as Goal hierarchies, enabling the tool to infer from the number of subgoals satisfied, the extent to which a general, parent goal has been achieved (functionally equivalent to a high level Criterion). Study 1 showed that a similar representation, Criterion trees, afforded a valuable representation when querying DRs (within the context of the retrieval tasks used). If DR tools recognised such abstractions, repetitiousness viscosity would be significantly reduced, for instance, the weighting of a whole set of bridging Criteria could be changed by changing their parent Criterion.

An example of knock-on viscosity is in changing decisions. If a new Option is selected, argumentation following on from the retracted decision needs to be changed or even deleted (e.g. other decisions need checking; a whole series of Questions becomes redundant). Another example of QOC dynamics is restructuring due to changing views of the design space. Updating can be made a less tedious process by automating recurring tasks. If the user can declare dependency relations between decisions and consequent DR, following a change, the tool can present dependent DR for the user to check and modify as necessary.

Different DR notations make provision for this kind of tool support to different degrees, as discussed earlier [§6.3.1.3]. However, it is worth noting that the representational *environment* can sometimes compensate for viscosity due to weaknesses in the notation it is running, by providing extra functionality. For instance a tool may still be able to differentiate two nodes with the same type, if it can combine a type descriptor (e.g. "Criterion") with a content descriptor like a keyword ("working memory" say). This is an example of the system using properties of the notation (entity types) and the environment (the facility to do textual search within nodes). Taking this one step further, Green (1989) provides instances from other domains of how the working environment can generate abstractions across information structures.

In sum, viscosity reduces as the representation makes more abstractions and dependencies explicit. Within the domain of DR specifically, 'fluidity' can be maintained when tools recognise issue and criterion hierarchies, and design dependencies for updating decisions.

10.6.1.3 Hidden dependencies

This dimension describes the extent to which a representation *hides* important (i.e. task-related) relationships between states. Everyday examples (Green et al, in press) where dependencies are only visible in one direction are the 'style' hierarchy in Microsoft™ Word ("which styles are parents to this one?"), the Macintosh™ Finder™ ("which folder is this folder in?"), and many spreadsheets ("which cells contribute to this one?").[8] The domain of design is rich in the dependencies which need to be managed, and in DR, these manifest in several ways:

Question dependencies

- a sequence of *consequent-Question* relationships communicates the problem decomposition (as in PHI's issue *serves* relation)

Criterion dependencies

- parent-child relationships exist between general and bridging Criteria—what happens if we increase the weight of Criterion X?

Option/Decision dependencies

- Options *depend-on* others (e.g. possible programming languages depend on the hardware platform)
- Options constrain others (e.g. adopting a particular metaphor may preclude Options elsewhere)

A *chain* of arguments is essentially a series of dependencies: if Argument *X* is invalidated, all arguments whose reasoning depends on *X* is invalidated.

The extent to which dependencies are hidden, is generally dependent on the environment, as opposed to the notation. The examples of viscosity management given in the previous section also serve as examples of dependencies which should not be hidden; so for

8 Interestingly, folder tracing information is now available in Macintosh™ System 7.0, and in Microsoft™ Excel 3.0, for instance, parent cells can be traced relatively easily.

example, if a tool can provide services which involve computing over inter-Criterion relationships, it should also be able to display those relationships if requested.

10.6.1.4 Role expressiveness

Role expressiveness refers to how clearly a "chunk" of information structure conveys its functional role. Programmers parse code in order to acquire a mental structure of the flow and transformation of information (*what happens where?*). Role expressive notations "display their plan structure clearly" (Green, 1989), such that the visual appearance of code can be used as a cue to identify its function. Thus, programmers are helped if they can quickly identify code with functional roles such as:

nested-loop
sort routine
printer call
documentation comment.

DR representations cannot be discussed in precisely the same terms, since (at least at present) they do not contain rules actively processing information. However, within a notation, there are recurring rhetorical structures which with experience come to be recognised as meaningful 'building blocks.' A role expressive DR notation answers the question "what kind of argumentation does this rationale structure represent?", where answers (depending on the notation) might be:

a trade-off between alternatives
a meta-discussion about the validity of a link
hierarchical decomposition of an issue into sub-issues
a pair of decisions dependent on another.

One would if possible seek salient structures to convey each of the above. Role expressiveness in graphical notations is conveyed not only by the role communicated by node/link types individually (e.g. this node is an *Option*), but also by the *structures* built from combining them. As a simple example, in QOC two Criteria trading off against two Options is a regularly recurring structure (Figure 10.3), and can be recognised as such without difficulty. This is an element of QOC's 'perceptual vocabulary' (Fitter and Green, 1979).

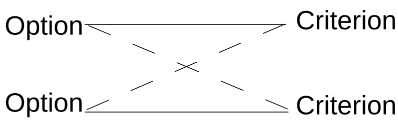


Figure 10.3: A role expressive structure in QOC, representing an Option trade-off.

If a notation places few constraints on how the content is to be structured (as with sketches and notes) it is much harder to identify functionally similar structures, simply because two structures fulfilling the same 'role' may be represented differently; the syntactic constraints imposed by DR notations reduce such variance. We see therefore how role expressiveness increases as we move from freeform text/sketches to semi-formal representation—a significant transition. However, is it possible to discriminate *between* DR notations?

Consider an example (from Lee and Lai, 1991b) illustrating role expressive differences between gIBIS and DRL. In gIBIS, a Position-Argument relationship (e.g. *supports*) can only be disputed indirectly by reusing existing node/link types, i.e. post a new Issue: *does Argument A really support Position P?* This is a question about the DR rather than the design itself, but notationally they both use the same Issue node type. In contrast, DRL provides distinct *Questions* and *Claims* nodes and *supports/objects-to* links for such debate. Thus, an *Issue-Position-Argument* structure is less role expressiveness than it would be if these constructs had a unique function: in order to ascertain their *roles* – how they relate functionally to the rest of the DR – it is necessary to read the content of the nodes.

So far, role expressiveness has been defined purely in terms of the salience with which entity types and structures convey their functional role. However, when one considers the power of 2D spatial layouts to express relationships, what we see is absolute and relative position taking on a special significance, introducing a new expressive dimension to entities. This is in essence a form of *implicit role expressiveness*. Raymond's (1991) work on the properties of visual notations, terms this *syntactic* and *semantic density*—elements' relative positions are not only infinitely variable, but have *meaning*.

Before concluding this section, a note of clarification is in order, to preempt confusion that "role expressiveness=meaningfulness." Initially, it seems reasonable to argue that a one-to-one mapping between form and function facilitates interpretation better than one-to-many, or many-to-one; indeed, for a machine this is valid. However, for people, there must be a stopping rule—when do we stop decomposing classes into finer categories? Consider two examples from the work with DRL (which has focussed most on enriching DR). The decision was taken not to differentiate between two link types *causal support* and *evidential support,* because the distinction was too fine for general usage. Lee also notes the range of possible *is-a-subGoal* relations, such as *mutually exclusive*, *overlapping*, and *exhaustive*. Users and situation of use must be weighed carefully if subtle differences such as these are to be introduced, and still be usable. In other words we must take into account the knowledge designers bring to the task; the usability of a notation rests critically on the mapping between the concepts with which designers work everyday, and the vocabulary offered by the notation. A functional role which to one person is clear may be to another quite opaque. Role expressiveness is a function of user expertise, which in the domain of DR means appreciating distinctions between the concepts represented by a notation's vocabulary.

10.6.2 Trade-offs between cognitive dimensions in DR authoring

The above analyses highlight the existence of trade-offs between dimensions. Whilst syntactic and semantic density add another dimension to role expressiveness which makes it extremely powerful on a local scale, there are well-known problems with using spatial layouts as the organising principle for large information structures (Card, 1989). There is also an interaction with viscosity. Structural changes to a representation based on implicit role expressiveness are easy to make, because there are no 'hard' links to break.[9] However, as information is massed, there is a parallel increase in the need for mechanisms to operate over the whole structure, which is of course where abstractions start to become useful.

One can now see the following interaction between dimensions: 'soft' structures relying on implicit role expressiveness minimise *local* viscosity (where the focus of attention is limited perhaps to restructuring DR over two or three screens); *global* viscosity (over a whole DR) cannot be tackled without abstractions, which in turn requires that entities be in a computable state (i.e. assigned a type); however, such global operations may not be needed until considerable amounts of material have accumulated, by which time the domain *may* be well enough understood that entities can be classified and linked without committing the designer prematurely to that structure. This constitutes as much a design hypothesis as a statement of fact, due to our impoverished understanding of the DR authoring process; the point is that the analysis is highlighting cognitive design trade-offs. Figure 10.4 summarises the trade-off between dimensions in the DR representational process.

9 In discussing the need for representational flexibility in hypertext tools, Marshall et al (1991) refer to a similar concept as "lightweight structure."

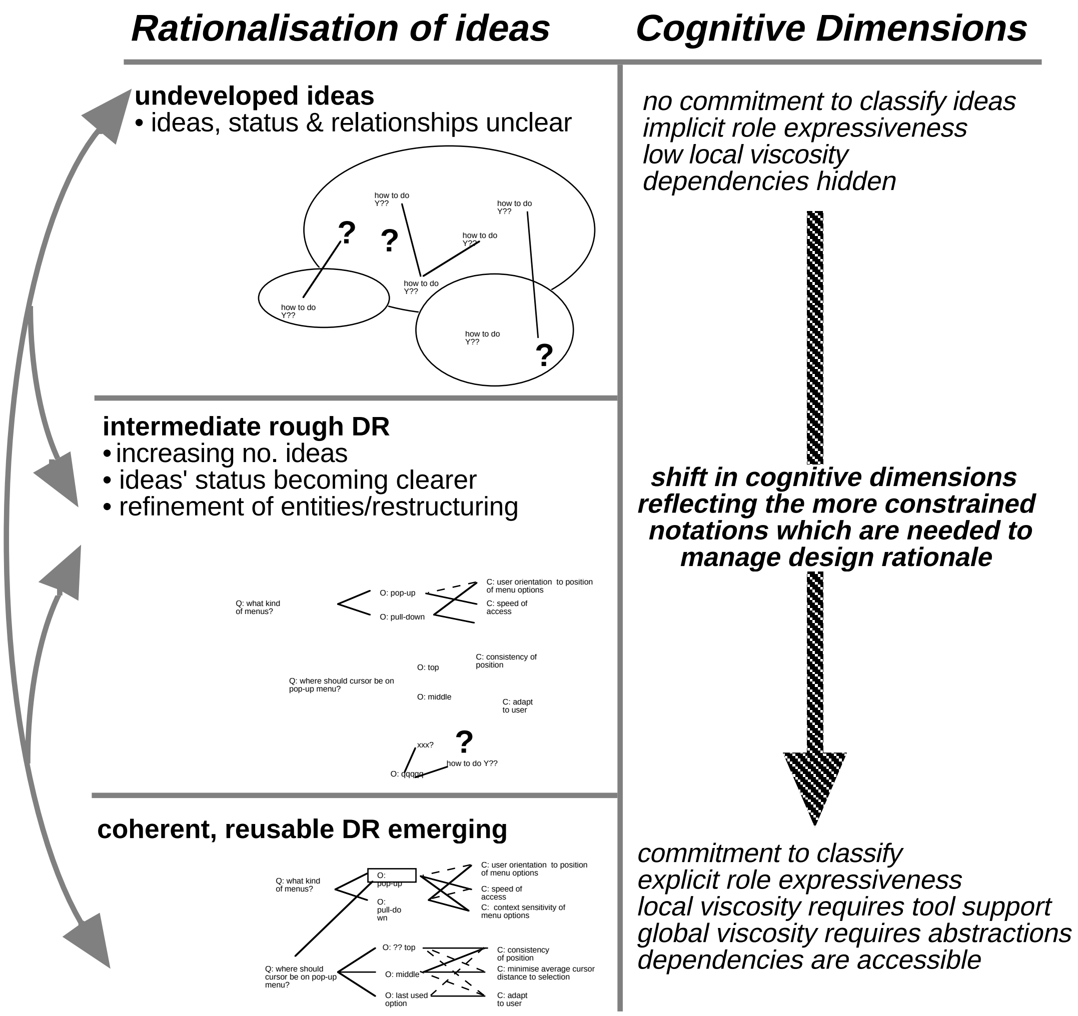


Figure 10.4: Trade-offs between cognitive dimensions at different stages in the process of rationalising argumentation. The curved arrows indicate that the process is non-linear: designers need to switch between representations opportunistically.

10.6.3 Conclusions on cognitive dimensions

It has been shown that once operationalised with respect to DR, cognitive dimensions map closely to many of the concerns in the QOC studies. The dimensions of *role expressiveness*, *viscosity*, *hidden dependencies*, and *premature commitment* have something positive to offer in this respect, making it possible to discuss "DR usability" in more precise terms. They serve to constrain the design space of DR authoring environments within boundaries which reflect our current understanding of the DR authoring process, highlighting task and user-centred design trade-offs.

10.7 Requirements for a QOC authoring environment

In the course of the studies reported, it became clear from specific requests, and observations of difficulties with QOC representations, that computational support has a key role to play if DR is to be a realistic prospect. Many of the requirements for tool support would be met by functionality which will be typical of the next generation of hypertext systems (Halasz, 1988; Marshall et al, 1991). Thus, all of the following could be used to good effect in a DR environment: virtual structures, structural search facilities, composite nodes, versioning, semi-formal structure within nodes, and the representation of multiple structures within a given DR (e.g. Goal hierarchies; Issue hierarchies; dependencies). Table 10.3 summarises tool requirements such as these, generated in Studies 1-3.

Recommended support	***Incident reported in:***
❑ computation of virtual structures by description (e.g. define Criterion as root node, and show all decisions it supports)	§3.2.5.1, §3.2.8
❑ fish-eye views with different functions	§3.2.7
❑ standard string search facility (most powerful when combined with structural search–below)	§3.2.6.1
❑ generic structure templates for searching	§3.2.6.2
❑ need to be able to query links for backing argumentation	§4.2.6.1
❑ need for QOC to be able to express dependency relationships	§4.2.3, §6.3.1.3
❑ visual coding of some sort for degrees of certainty (indicating status using a categorical scale)	§4.2.4.3
❑ links need to be maintained between a Criterion and its other instances; the Criterion tree is one way to achieve this	§3.3.1 §4.2.3
❑ need to interface brainstorming, 2D drawing tool interfaces seamlessly with hypertext functionality	§10.6.1.1

Table 10.3: Summary of main QOC tool requirements generated in Studies 1-3.

The prime motivation for seeking to understand the demands and modes of working with QOC authoring is to constrain the design of DR tools within boundaries defined by user and task-oriented factors. Whilst the specific requirements above are important, they are set in context under the following three ‘requirements-themes’ which have broad implications for shaping the design of notations and support environments.

10.7.1 Designing DR environments for restructuring

One of the results from studying DR authoring is that representational tasks begin to be recognised which arise as a function of the process of design deliberation. *Re*structuring activity is perhaps the best example. Often, the status of ideas only becomes clear as they are represented and discussed, creating the need to modify the structure [§6.2.5, §10.6.1.2]. The easier it is to make changes (i.e. the less viscous it is), the more likely designers are to use the tools. Although the content of nodes must be left to the designer, tools should at least partially automate common restructuring operations [§7.2.1.2] to reflect shifts in viewpoint. Operations which were observed included decomposing a general Criterion into more specific Criteria, grouping two Criteria into one more general Criterion and updating the Assessments, or moving an Option up or down a level of abstraction. It is hypothesised that further study of QOC use will show these to be generic operations.

10.7.2 Finding the balance in the size of notational vocabulary

The importance of the trade-off between computational power and usability was highlighted in the introduction to DR. The QOC studies demonstrate not only the sorts of initial errors and typical representational weaknesses encountered with even a simple notation, but also that some designers learnt to use QOC effectively and creatively. Empirical studies are needed to show how the more expressive notations like DRL, or enhanced versions of gIBIS (e.g. Conklin, 1989, Lubars, 1989) affect the deliberation process. Three independent variables of relevance here are *when* it is used, *by whom*, and *for what kinds of design*.

One consideration is how the developers envisage the tool being used. When DR is to be captured during meetings, the emphasis is on fewer, more general types (e.g. IBIS or QOC) . As Lee (1991) points out, for a tool like SIBYL not intended for use in real time, there is more time to choose between more complex and subtle differences in entity-relationship types.

In order to tailor DR to different user groups with different information requirements and expertise, different node types can be added or filtered out. However, one might also wish to consider the possibility that different levels of refinement for a given node/link type could be controlled. For example, in representing a particularly complex series of decisions, the designers might have to use quite fine distinctions between relationships (*causal support* and *evidential support*, for instance); however, in order to present such a rationale to a less expert user group it would be convenient to replace these with a more general *supports* link. The level of the constructs would thus be tailored to the user's knowledge.

One concludes from the studies reported here that developers should think hard each time they wish to add another entity or relation type to the core elements of a notation.

10.7.3 Argumentation alone is not enough: learning from writing tools

A line of research close to DR tool development is the work on writing support tools, reviewed in §1.3.2. It was noted that a noticeable characteristic of all these tools is that they share an emphasis on providing *multiple representations* of the ideas to be organised, covering the whole representational process from initial ideas to rationalisation (e.g. unconnected notes, loose networks, argumentation structures, hierarchies, linear outlines). It is accepted without question that authors employ different cognitive representations at different phases in the authoring process. In stark contrast, whilst DR tools may allow ideas encoded as argumentation to be viewed in different ways, they provide no support to the author in getting to that stage initially.

Whilst there are clearly important research issues remaining in writing cognition research, Sharples and Pemberton (1988) characterise the "consensus model" of writing which has been consolidated over a decade's research, summarised below:

- Writing models must cover the whole process of writing, from initial mustering of ideas, to final presentation on paper.
- The act of writing triggers new insights, leading to new intentions, and plan revision.
- There are many strategies to writing. A tool should enable writers to choose, and subsequently support the process which best fits their mode of working.
- Writing is managed by setting constraints; writers can be taught well-defined techniques of constraint management.
- For teaching purposes, a meta-language is needed to describe writing operations, strategies, and approaches to structuring plans and texts.

One problem for builders of DR environments is that there are at present no cognitive models of DR authoring. Essentially, DR has yet to catch up with writing research in studying the ability of its subjects (designers) to work with the representations which the tools use. Whilst this is not surprising given the short time that DR has been even a recognisable field of research, further work is needed to rectify this imbalance. The research reported in this thesis does, however, provide a starting point. If we consider the generic writing model summarised above, the evidence suggests that DR authoring displays many of its characteristics (listed in the same order):

- The DR authoring process moves from the relatively unstructured mass of ideas in initial idea generation and debate, to a crafted structure which organises those ideas to different degrees (depending on the DR approach). A DR environment should provide seamless support for the whole process.
- The process of reifying reasoning as QOC led designers to reformulate the initial expression of ideas in several beneficial ways, summarised earlier §10.2. The opportunistic switching between subtasks, often between different areas of the DR structure, should be supported within the environment.
- The designers adopted different, sometimes explicitly declared strategies in constructing QOCs, depending on which ideas suggested themselves first. Tool developers should be wary of imposing sequencing constraints on designers (although clearly this will trade off against the system's need to maintain a consistent model of the structure).
- The 'styleguide' presented in §10.3 is an initial suggestion as to what 'well-formedness' might mean in the context of QOC. The aim of such representational

constraints is to assist designers in conducting accurate, elegant, succinct design space analyses, as well as enhance QOC as a vehicle for communicating reasoning.

With respect to the last feature of the writing consensus model, it is not clear how useful a meta-language for describing authoring would be in the context of DR. A meta-language would describe different strategies for constructing QOCs, such as *decompose Questions hierarchically first*, or *define Criterion tree first*, or *generate Options first, then Questions*. A more complex strategy would be to follow a process model such as the one evaluated in Case Study 1. In terms of tool support, the process model could be explicitly modelled in much the same way that Cognoter's and Argnoter's functionality were constrained according to the phases of explicit process models (Stefik et al, 1987– reviewed in §1.3.1.4).

The results of this thesis constitute the basis of an informal cognitive model of DR authoring. As with writing models, the model motivates the development of an environment to support the smooth transition from the *informal* (unclassified ideas expressed at varying levels of abstraction) to the *semi-formal* (typed entities and relations at a consistent level), and from the incompleteness and inconsistencies of *initial reasoning*, to the rigour and coherence of *retrospective reasoning*.

10.8 WHERE NARRATIVE AND RATIONALISED DR MEET

One of the clearest analyses of issues facing DR research is that put forward by Conklin and Yakemovic (1991), identifying the process-oriented vs. structure-oriented dimension. Whilst this distinction is derived primarily from what the DR offers *once created* (a logical or chronological record of reasoning), the process-oriented approach as presented is also strongly biased toward a particular *way* of authoring chronological DR, namely DR capture 'as it happens' in meetings.

Despite the emphasis on IBIS as a rhetorical model for meetings, in their field study of itIBIS in use, Conklin and Yakemovic record that "itIBIS... turned out to be especially useful when used to capture notes during meetings or to restructure them afterwards." Whilst there is no indication of the extent to which itIBIS was used in each way, or which method produced the most useful DR, what seems to be claimed is that process-oriented DR can in fact be created during *or* after the deliberation process.

If we view approaches to DR in terms of the two dimensions of *when* the DR is created versus *what kind of DR* it presents, the space might be represented as in Figure 10.5.

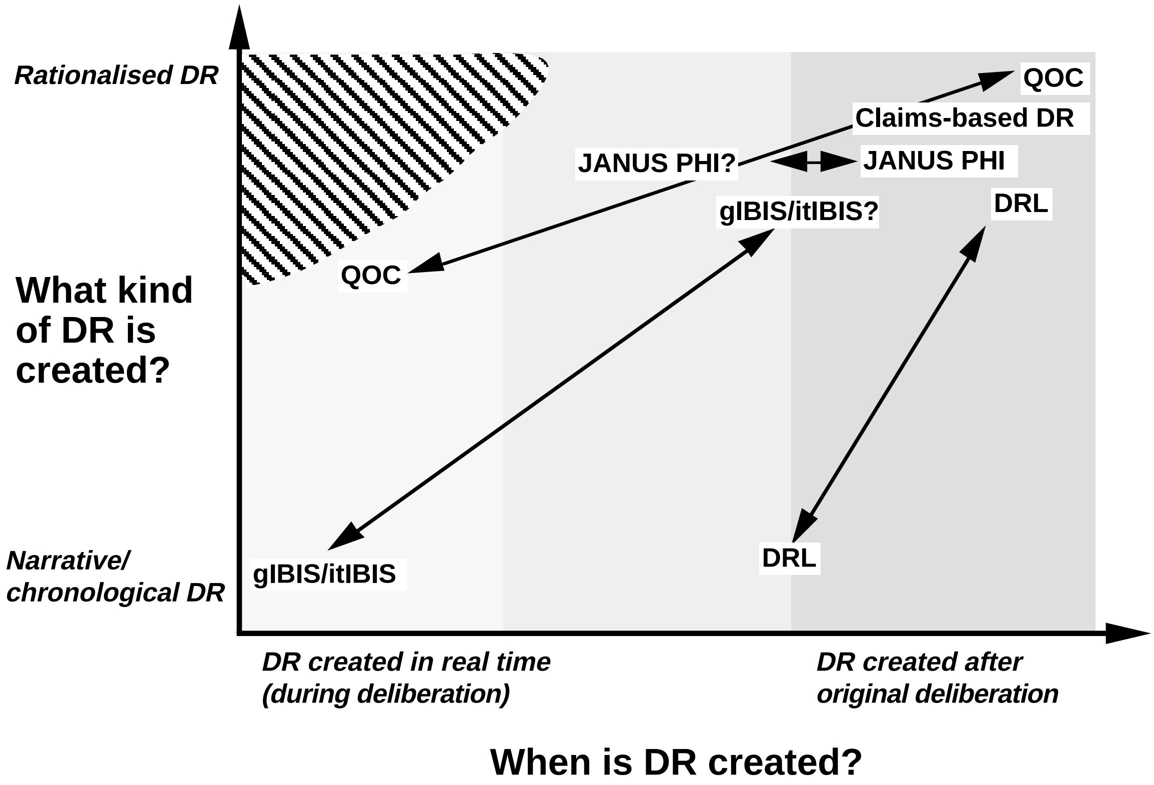


Figure 10.5: Locating DR approaches within a space, with axes defined by when the DR is created, versus the kind of DR created.

The shaded triangle reflects the fact that it is extremely difficult to generate rationalised DR as the ideas are being expressed for the first time: there is inevitably a delay whilst the implications of the ideas are realised, equivalent to deciding how to phrase and structure the new DR. Some notations are shown as lying along specific paths within the space. This reflects the fact that they can be used at different points relative to deliberation, which invariably means capturing different kinds of DR.

DRL's larger notation makes it possible to represent debate about most aspects of design. The Claims and counter-Claims which can be made on behalf of entities and relations lend themselves naturally to conveying narrative rationale about local issues in a process-oriented way. However, DRL inter-issue relations are overridingly non-historical – a DRL graph does not grow over time in the same way that a gIBIS graph grows. Lee (1991) also expresses the clear view that SIBYL is best used asynchronously, the extra time which SIBYL users have to weigh their contributions making it possible to build the DR in a more logical than chronological fashion.

IBIS is shown in its expected position as a narrative structure authored in real time. However, as already noted, Conklin and Yakemovic also hint that gIBIS can be used in a structure-oriented way; furthermore, referring to the structural and temporal views of DR, they state that:

> Sorting out how to separate – and then smoothly integrate – these two views is a major challenge for our research.

It is almost inevitable that reconstructing a meeting will produce more rationalised DR than notes taken as the meeting unfolds (unless, for instance, the retrospective DR is created from audio/video recordings). Use of gIBIS in this way brings it very close to the DSA mode of working.

The QOC profile covers virtually the whole temporal dimension, and a subsection of the content dimension. This captures the authoring process of reorganising the space over an extended period to capture new insights into its underlying structure. Clearly, when initially recording QOC, the designer is engaging in a task very similar to using gIBIS. When trying to initially translate ideas into semi-formal structure, the somewhat subtle differences between Issues and Questions, and between Arguments and Criteria are secondary. However, whereas the process-oriented gIBIS user engages in very little or no revision, the QOC user continues to refine the representation, searching for better Questions.

The claims-based approach is perhaps the most clearly rationalised and abstracted representation of DR. There is no attempt to capture the original design process in any way, since the DR analysts are not the original designers.

JANUS sits somewhat awkwardly within this space, as the PHI issue bases are not actually constructed in relation to a specific design session; rather, they are generic issue-bases, made available to designers. They are shown as being created retrospectively in Figure 10.5 to capture the fact that considerable engineering effort is involved on the part of the DR's original authors to prepare the issue-base (the considerable number of issues; identifying the generic issues; linking to the critics).

The analysis of QOC use using Tang's framework [§10.4] drew the distinction between expressing ideas through the representation, or using it purely for storing ideas. That is to say, DR can shape design discussion (and hence, to some extent, the design) to a greater or lesser extent depending on the approach. At one end of the continuum, equivalent to the *highly rationalised* (top right) corner of Figure 10.5, is 'strong DSA' which is the most directive approach; this can then be weakened in degrees to reduce the authoring overheads, as shown in Figure 10.6.

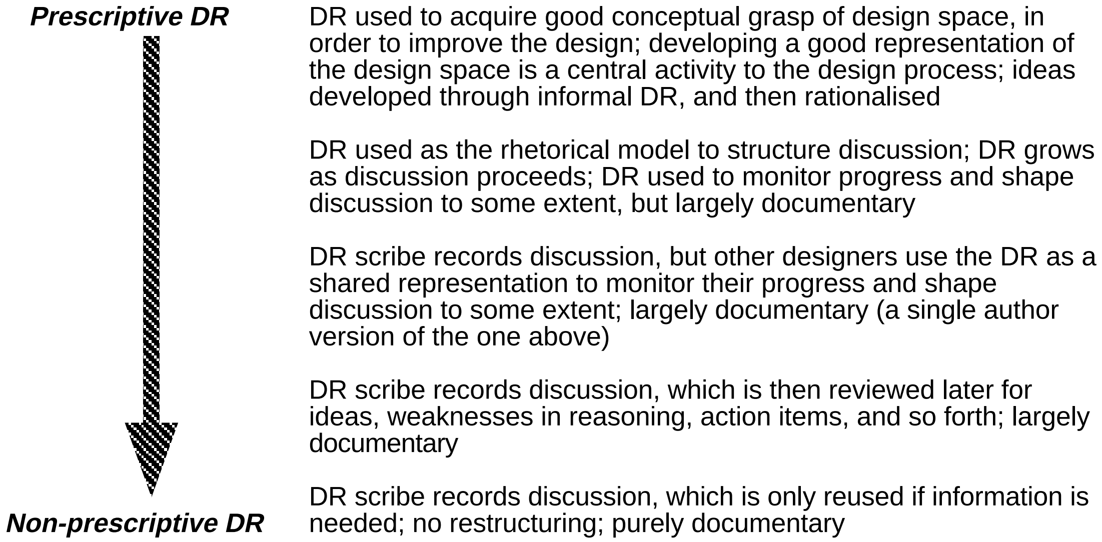


Figure 10.6: Ways in which DR can be used, lying along a continuum of the contribution it makes to deliberation (or conversely, the overheads involved in learning to use).

What conclusions can be drawn? The *when–what* space demonstrates where process-oriented and structure-oriented approaches meet. The evidence from Study 3 and Case Study 1 strongly suggests that designers cannot be expected to learn DSA as an approach to design before they are competent with using QOC at a more 'basic' level – designers in these studies translated ideas without undue difficulty, *recorded* discussion using QOC (i.e. as with gIBIS), and sought to refine Questions – to a limited extent. However, DSA heuristics which introduced radically different tasks from normal design (such as *consider extreme Options*) were not used, and nor, one suspects, would have been global Criteria such as *search for generic Questions*. In sum, it could be argued that designers in these studies were using QOC in a fashion very close to process-oriented gIBIS. Initially, learning DSA with QOC is much the same as learning to capture discussion with gIBIS. It may only be when some expertise has developed that the differences emerge, and an awareness of the DSA approach drives the topics for discussion, as opposed to passively recording it.

10.9 AVENUES FOR FUTURE RESEARCH INTO DR USABILITY

The key themes to have emerged from this research have been drawn together over the preceding sections. A number of open research issues are now briefly considered.

Firstly, it is hoped that the studies reported here will stimulate more empirical research into how designers make use of DR notations and tools. The primary variables determining the usability and acceptability of DR are the notation, the tool environment in which it runs, the designers' expertise, the methodology – if any – for using the notation, and the organisational culture. An almost inexhaustible list of research questions is associated with the properties of, and interactions between these factors, of which a few are: How generalisable are the phenomena observed with QOC use? What evidence is there that QOC aids communication, or improves design reasoning? How easy is it to recover reasoning using gIBIS? What are the overheads in supporting DR reuse with JANUS or SIBYL? How much expertise is needed before designers can benefit from conducting a claims analysis? For what classes of design are argumentation-based notations best suited? How prescriptive should methodologies be? How would a project's use of DR over an extended period change the design process?

Secondly, but essentially unpacking the variable of tool support mentioned above: Are the recommendations for tool functionality valid? Compared to non-computational media, a tool's representational power comes at the cost of an additional gulf of translation and execution over and above the *notation*, namely, the tool's *user-interface*. Moreover, as tool developers strive for ever greater representational mutability, it remains to be seen if DR, once reified, can sufficiently match the cognitive transformations which designers exercise in their reasoning and conceptualisations of the design space.

A fundamental question is, *What kind of DR do we want to represent?* There is nothing gained in developing notations, methods and tools to represent redundant knowledge. It was suggested in Study 1 that one approach to answering this is to survey potential DR users in order to build a taxonomy of the classes of information which a DR could provide. Thus, a third avenue for research might take the form of field studies of system maintenance or redesign communities—what kinds of questions do they really ask?

Finally, an intriguing area for development is the integration of the argumentative approach to design and its representation, with other schools of thought in design research. An initial step in this direction was the use of an existing framework for characterising representational activity at the *group* level to describe use of DR [§10.4]. The ethnomethodological approach to design research (e.g. Bucciarelli, 1988), looking at, for instance, social processes in design practice (Minneman, 1991) represents a perspective with which DR-oriented approaches to design must reconcile themselves. Whether DR notations or frameworks can provide a language or medium via which the social dynamics of design can be *mediated*, or whether they are incompatible with such processes, is a problem for both empirical and analytic effort, answers to which will assist in weaving together strands of design research which are currently separate.

10.10 CONCLUDING THOUGHTS

Despite the fact that its conceptual roots trace back to the 1960s, DR research as a field in its own right is just beginning. There are many good reasons for representing design reasoning in an explicit, computable form, with its potential to support design reasoning, manage dependencies, and assist in building cumulative design knowledge. However, at present, it is far from clear whether DR will gain general acceptance. As with any new idea, a critical mass of support is needed, up until which point the concepts, formalisms, and tools remain the province of the research literature.

Currently, a few research groups are charting DR's design space, exploring the key dimensions of expressiveness, computability, and usability. What is needed is conceptual and empirical work to stretch the boundaries of this triangle in order to discover when it breaks. At what point does a highly expressive notation become unusable? What is the simplest notation which affords useful information, and for which kinds of design? A fourth variable which will become increasingly important as more tools are developed is the quality of the user interface—innovative, low cost ways for recording, retrieving, and reusing rationale should make the triangle more 'elastic'—increasing expressiveness and computability without sacrificing usability.

This thesis is a step towards describing the phenomenon with which we are concerned. It is necessary to know in precise terms what it is we are presenting to software designers before we can make informed claims about the overheads and benefits involved. It is vital to understand the relationship between designers' cognitive representations and the representations we expect them to create and use, particularly in the context of a fast-flowing, time-pressured activity such as software design, in which 'documentation' is already a bad word. The research reported here has explored the learnability and usability of a particular design rationale notation, with respect to its creation and retrieval. This work should facilitate the process of discovering if, and how, design rationale can be integrated within the software design process.

References

Adams J.K. and Adams P.A. (1961) Realism of confidence judgements. *Psychological Review*, 68, 33-45

Adelson B. and Soloway E. (1984) A cognitive model of software design. *Technical Report 342,* Department of Computer Science, Yale University

Anderson J.R. (1983) *The Architecture of Cognition*. Harvard University Press: Cambridge, Massachusetts

Balzer R., Cheatham T.E. and Green C. (1983) Software technology in the 1990s: Using a new paradigm. *IEEE Computer*, 16 (11), 39-45

Barlow J., Beer M., Bench-Capon T., Diaper D., Dunne P.E.S. and Rada R. (1989) Expertext: Hypertext-expert system theory, synergy and potential applications. In Shadbolt N. (ed.) *Research and Development in Expert Systems VI*, 116-127. Cambridge University Press: Cambridge

Bellotti V.M.E., MacLean A. and Moran T. (1991) Structuring the design space by formulating appropriate design rationale questions. *SIGCHI Bulletin*, 23 (4), 80-81. Also *Amodeus Working Paper RP6/WP6,* Rank Xerox EuroPARC

Black A. (1990) Visible planning on paper and on screen: The impact of working medium on decision-making by novice graphic designers. *Behaviour and Information Technology,* 9 (4), 283-296

Bobrow D.G. and Goldstein I. (1980) Representing design alternatives. *Proceedings of the Conference on Artificial Intelligence and the Simulation of Behaviour*, Amsterdam, July 1980. Reprinted in Goldstein I. and Bobrow D.G. (1981) An experimental description-based programming environment: Four reports. *Technical Report CSL-81-3*, Xerox Palo Alto Research Center

Bobrow D.G., Stefik M., Foster G., Halasz F., Lanning S. and Tatar D. (1990) The Colab Project final report. *Technical Report SSL-90-45*, Xerox Palo Alto Research Center

Booch, G. (1991) *Object Oriented Design with Applications.* Benjamin/Cummings

Boose J.H., Shema D.S. and Bradshaw J.M. (1990) Capturing design knowledge for engineering trade studies. In Wielinga J., Boose J.H., Gaines B., Schreiber G. and Van Someren (eds.) *Current Trends in Knowledge Acquisition*, 78-89. IOS Press: Amsterdam

Bouldin (1989) *Agents of Change: Managing the Introduction of Automated Tools.* Yourdon Press: Englewood Cliffs, NJ

Botterill J.H. (1982) The design rationale for the System/38 user interface. *IBM Systems Journal*, 21 (4), 384-423.

Bradshaw J.M. and Boose J.H. (1991) Mediating representations for knowledge acquisition. *Proceedings of Knowledge Acquisition: From Techniques to Tools Workshop*, AAAI, July 1991

BrockreideW. and Ehninger D. (1960) Toulmin on argument: An interpretation and application. *Quarterly Journal of Speech*, 46, 44-53

Brown J.S. (1982) Notes concerning desired functionality, issues and philosophy for an AuthoringLand. *CIS Working Pape*r, Xerox Palo Alto Research Center

Brown J.S. (1983) Process versus product: A perspective on tools for communal and informal electronic learning. *Report from the Learning Lab: Education in the Electronic Age*, by the Educational

Broadcasting Corporation; reprinted in *Journal of Educational Computing Research*, 1 (2), 179-202 (1985)

Brown J.S. (1986) From cognitive ergonomics to social ergonomics and beyond. In Norman D.A. and Draper S.W. (eds.) *User Centered System Design*, 457-486. Lawrence Erlbaum Associates: Hillsdale, New Jersey

Bruns G. (1988) GERM: A metasystem for browsing and editing. *Technical Report STP-122-88*, Microelectronics and Computer Technology Corporation

Bucciarelli L.L. (1988) An enthnographic perspective on engineering design. *Design Studies*, 9 (3), 159-168

Burleson B.R. (1979) On the foundations of rationality: Toulmin, Habermas, and the 'a priori' of reason. *Journal of the American Forensic Association*, 16, 112-127

Bush V. (1945) As we may think. *Atlantic Monthly*, 101-108

Card S.K. (1989) Information workspaces. *Friend 21 Conference: International Symposium on Next Generation Human Interface Technologies*, Sept. 5-6, Tokyo

Card S.K., Robertson G.G. and Mackinlay J.D. (1991) The Information Visualiser, an information workspace. *Proceedings of CHI'91: Human Factors in Computing Systems*. ACM: New York

Carey T.T., Hunt W.T. and López-Suárez A. (1991) Roles for tables of contents as hypertext overviews. In Diaper D., Gilmore D., Cockton G. and Shackel B. (eds.) *Human-Computer Interaction: Proceedings of Interact'90*, 581-586. Elsevier Science Publishers: N. Holland

Carey T.T., McKerlie D.L., Bubie W. and Wilson J. (1991) Communicating human factors expertise through design rationales scenarios. In Diaper D. and Hammond N.V. (eds.) *People and Computers VI*, 117-130. Cambridge University Press: Cambridge

Carroll J.M. and Campbell R.L. (1989) Artifacts as psychological theories: The case of human-computer interaction. *Behaviour and Technology*, 8 (4), 247-256

Carroll J.M. and Kellogg W.A. (1989) Artifacts as theory nexus: Hermeneutics meets theory-based design. *Proceedings of CHI'89: Human Factors in Computing Systems*. ACM: New York

Carroll J.M., Kellogg W.A. and Rosson (1991) The task-artifact cycle. In Carroll J.M. (ed.) *Designing Interaction: Psychology at the Human-Computer Interface*. Cambridge University Press: New York

Carrroll J.M., Miller L.A., Thomas J.C. and Friedman H.P (1980) Aspects of solution structure in design problem solving. *American Journal of Psychology*, 93 (2), 269-284

Carroll J.M. and Rosson M.B. (1985) Usability specifications as a tool in iterative development. In Hartson H.R. (ed.) *Advances in Human-Computer Interaction*. Ablex: Norwood, New Jersey

Carroll J.M and Rosson M.B. (1987) The paradox of the active user. In Carroll J.M. (ed.) *Interfacing Thought: Cognitive Aspects of Human-Computer Interaction*, 80-11. MIT Press/Bradford Books: Cambridge, Massachusetts

Carroll J.M and Rosson M.B. (1990) Human-computer interaction scenarios as a design representation. *Proceedings 23rd Hawaii International Conference on System Science*, 533-539. IEEE Computer Society Press

Carroll J.M and Rosson M.B. (1991a) Deliberated evolution: Stalking the View Matcher in design space. *Human-Computer Interaction*, 6 (3&4), 281-318

Carroll J.M and Rosson M.B. (1991b) Getting around the task-artifact cycle: How to make claims and design by scenario. *Technical Report*, IBM Watson Research Center

Cross N. (1990) The nature and nurture of design ability. *Design Studies*, 11 (3), 127-140

Conklin J. (1987) Hypertext: An introduction and survey. *IEEE Computer, 20(9), 17-41*

Conklin J. (1989a) Design rationale and maintainability. *Proceedings 22nd Hawaii International Conference on System Science*, 555-561. IEEE Computer Society Press

Conklin J. (1989b) Interissue dependencies in gIBIS. *Technical Report STP-091-89*, Microelectronics and Computer Technology Corporation

Conklin J. and Begeman M.L. (1989) gIBIS: A tool for all reasons, *Journal of the American Society for Information Science*, (May), 200-213

Conklin J. and Yakemovic K.C.B. (1991) A process-oriented approach to design rationale. *Human-Computer Interaction*, 6 (3&4), 357-391

Consens M.P. and Mendelzon A.O. (1989) Expressing structural hypertext queries in GraphLog. *Proceedings of Hypertext'89*, 269-292. ACM: New York

Cooley J.C. (1959) On Mr. Toulmin's revolution in logic. *Journal of Philosophy*, 56, 297-319

Cowan J.L. (1964) The uses of argument – an apology for logic. *Mind*, 73, 27-45

Curtis B., Krasner H. and Iscoe N. (1988) A field study of the software design process for large systems. *Communications of the ACM*, 31 (11), 1268-1287

Delisle N. and Schwartz (1986) Neptune: A hypertext system for CAD applications. *Proceedings of International Conference on Management of Data*, 132-143. ACM: New York

de Souza F. and Bevan N. (1990) The use of guidelines in menu interface design: Evaluation of a draft standard. In Diaper D., Gilmore D., Cockton G. and Shackel B. (eds.) *Human-Computer Interaction: Proceedings of Interact'90*, 345-440. Elsevier Science Publishers: N. Holland

Eder W. E. (1990) Design science—meta-science to engineering design. *Proceedings of the 2nd International Conference on Design Theory and Methodology*, 327-338. Chicago

Elworthy D. (1991) Analysis of the Design Rationale Tutorial. *EuroPARC Working Paper*, October 1991

Englebart D.C. (1963) A conceptual framework for the augmentation of man's intellect. In *Vistas in Information Handling*, Vol.1. Spartan Books: London

Englebart D.C. and English W.K. (1968) A research center for augmenting human intellect. *AFIPS Conference Proceedings*, Vol.33, Part 1. The Thompson Book Company: Washington DC

Evans J. and Wason P.C. (1976) Rationalisation in a reasoning task. *British Journal of Psychology*, 63, 205-212

Fairchild K.M., Meredith G. and Wexelblat A. (1988) The Tourist artificial reality. *Technical Report STP-310-88*, Microelectronics and Computer Technology Corporation

Fairchild K.M., Poltrock S.E. and Furnas G.W. (1987) SemNet: Three-dimensional graphic representations of large knowledge bases. In Guindon R. (ed.) *Cognitive Science and its Applications for Human-Computer Interaction*. Lawrence Erlbaum Associates: Hillsdale, N.J

Fikes R. (1988) Integrating hypertext and frame-based domain models. *AAAI 1988 Workshop on AI and Hypertext*, St. Paul, Minnesota

Finger S. and Dixon J.R. (1989) A review of research in mechanical engineering design. Part I: Descriptive, Prescriptive and Computer-based models of design process. *Research in Engineering Design*, 1 (1), 51-67

Fischer G. (1988) A critical assessment of hypertext systems. *Panel Session in Proceedings of CHI'88: Human Factors in Computing Systems*, 223-227. ACM: New York,

Fischer G., Lemke A.C. and Morch A.I. (1989) JANUS: Integrating hypertext with a knowledge based design environment. *Proceedings of Hypertext'89*, 105-117. ACM: New York

Fischer G., Lemke A.C., McCall R. and Morch A.I. (1991) Making argumentation serve design. *Human-Computer Interaction*, 6 (3&4), 393-419

Fitter M. and Green T.R.G. (1979) When do diagrams make good computer languages? *International Journal of Man-Machine Studies*, 11, 235-261

Furnas G.W. Generalised fisheye views. *Proceedings of CHI'86: Human Factors in Computing Systems*, 16-23. ACM: New York

Gammack J.G. (1987) Modelling expert knowledge using cognitively compatible structures. *Proceedings of 3rd International Conference on Expert Systems*, 191-200. Learned Information: London

Ghiselli E.E., Campbell J.P. and Zedeck S. (1981) *Measurement Theory for the Behavioural Sciences.* Freeman: San Francisco

Goel V. and Pirolli P. (1989) Motivating the notion of generic design within information-processing theory: The design problem space. *AI Magazine*, Spring 1989, 18-36

Goldstein I.P. and Bobrow D.G. (1980) Representing design alternatives. *Proceedings of the Conference on Artificial Intelligence and the Simulation of Behaviour*, Amsterdam, July, 1980. Reprinted in Goldstein I.P. and Bobrow D.G. (1981) An experimental description-based programming environment: Four reports, *Technical Report CSL-81-3*, Xerox Palo Alto Research Center

Green T.R.G. (1982) Pictures of programs and other processes, or how to do things with lines. *Behaviour and Information Technology*, 1 (1), 3-36

Green T.R.G. (1989) Cognitive dimensions of notations. In Sutcliffe A. and Macaulay L. (eds.) *People and Computers V*, 443-460. Cambridge University Press: Cambridge

Green T.R.G. (1990) The cognitive dimension of viscosity: A sticky problem for HCI. In Diaper D., Gilmore D., Cockton G. and Shackel B. (eds.) *Human-Computer Interaction: Proceedings of Interact'90*, 79-86. Elsevier Science Publishers: N. Holland

Green T.R.G., Bellamy R.K.E. and Parker J.M. (1987) Parsing and gnisrap: A model of device use. In Olson G.M., Sheppard S. and Soloway E., *Empirical Studies of Programmers: Second Workshop*. Ablex: Norwood, New Jersey

Green T.R.G., D.J. Gilmore, B.B. Blumenthal, S. Davies and R. Winder (in press) Towards a cognitive browser for OOPS. *International Journal of Human-Computer Interaction*

Greenberg S. and Bohnet R. (1991) GroupSketch: A multi-user sketchpad for geographically-distributed small groups. *Proceedings of Graphics Interface'91*

Grudin J. (1991) Systematic sources of sub-optimal interface design in large product development organisations. *Human-Computer Interaction*, 6 (2), 147-196

Guindon R., Curtis B. and Krasner H. (1987) A model of cognitive processes in software design: An analysis of breakdowns in early design activities by individuals. *Technical Report STP-283-87*, Microelectronics and Computer Technology Corporation

Guindon R. (1990) Designing the design process: Exploiting opportunistic thoughts. *Human-Computer Interaction*, 5, 305-344

Gullichsen E., D'Souza P., Lincoln P. and Casey T. (1986) The PlaneText Book. *MCC Technical Report STP-333-86(P),* Microelectronics and Computer Technology Corporation

Hair D.C. and Lewis C. (1990) Are argumentation representation schemes useful? *Technical Report CU-CS-475-90,* Department of Computer Science, University of Colorado at Boulder, Boulder, Colorado 80309-1592

Halasz F.G. (1988) Reflections on NoteCards: Seven issues for the next generation of hypermedia systems. *Communications of the ACM*, 31, 836-852

Halasz F.G., Moran T.P. and Trigg R.H. (1987) NoteCards in a nutshell. *Proceedings of CHI and GI'87: Human Factors in Computing Systems and Graphic Interface*, 45-52. ACM: New York

Halasz F.G. and Schwartz M. (1990) The Dexter hypertext reference model. *Proceedings of the Hypertext Standardisation Workshop*, National Institute of Standards and Technology, 95-133 (available as NIST special publication 500-178, March 1990)

Hammond N.V., Jørgensen A., MacLean A., Barnard P. and Long J. (1983) Design practice and interface usability: Evidence from interviews with designers. *Human Factors Laboratory Report 082*, IBM UK, Hursley Park.

Heidegger M. (1962) *Being and Time* (translated by John Macquarrie and Edward Robinson). Harper and Row: New York

HCI (1991) Special Issue on Design Rationale, *Human-Computer Interaction*, 6 (3 & 4).

Hutchins E. and Norman D.A. (1988) Distributed cognition in aviation. *Unpublished manuscript*. Department of Cognitive Science, University of California, San Diego

Hypertext'89 Panel Session (1989) Expert systems and hypertext. *Proceedings of Hypertext'89*, 391-392. ACM: New York

Ishii H. (1990) TeamWorkStation: Towards a seamless shared workspace. *Proceedings CSCW'90: Computer Supported Cooperative Work*, 13-26. ACM: New York

Jansson D.G. and Smith S.M. (1991) Design fixation. *Design Studies*, 12 (1)

Jeffries R., Turner A.A., Polson P. and Atwood M.E. (1981) The processes nivolved in designing software. In Anderson J.R. *Cognitive Skills and Their Acquisition*, 255-283. Lawrence Erlbaum Associates: Hillsdale, New Jersey

Johnson J. and Beach R.J. (1988) Styles in document editing systems. *IEEE Computer*, 21(1), 32-43

Jordan D., Russell D.M., Jensen A-M.S. and Rogers R.A. (1989) Facilitating the development of representations in hypertext with IDE. *Proceedings of Hypertext'89*, 93-104. ACM: New York

Krasner H., Curtis B. and Iscoe N. (1987) Communication breakdowns and boundary spanning activity on large programming projects. In Olson G.M., Sheppard S. and Soloway E., *Empirical Studies of Programmers: Second Workshop*. Ablex: Norwood, New Jersey

Kunz W. and Rittel H.W.J. (1970) Issues as elements of information systems. Technical Report S-78-2, Institut für Gundlagen der Planung i.A, Universität Stuttgart, Kleperstraße 11, 7000 Stuttgart 1, Germany. Available as *Institute of Urban and Regional Development, Working Paper 131*. University of California, Berkeley

Lai K-Y., Malone T. and Yu K-C. (1988) Object Lens: A “spreadsheet” for cooperative work. *ACM Transactions on Office Information Systems*, 6 (4), 332-353

Lee J. (1990) SIBYL: A qualitative decision management system. In P. Winston and S. Shellard (eds.), *Artificial Intelligence at MIT: Expanding Frontiers*, 105-133. MIT Press: Cambridge, Massachusetts

Lee J. (1990) SIBYL: A tool for managing group decision rationale. *Proceedings CSCW’90: Computer Supported Cooperative Work,* Los Angeles. ACM: New York

Lee J. (1991) Extending the Potts and Bruns model for recording design rationale. *Proceedings 13th International Conference on Software Engineering,* Austin, Texas.

Lee J. and Lai K. (1991a) A comparative analysis of design rationale representations. *Center for Coordination Science Technical Report 121,* Massachusetts Institute of Technology

Lee J. and Lai K. (1991b) What’s in design rationale? *Human-Computer Interaction*, 6 (3&4), 251-280

Lewis C., Rieman J. and Bell B. (1991) Problem-centred design for expressiveness and facility in a graphical programming system. *Human-Computer Interaction*, 6 (3&4), 319-355

Lichtenstein S., Fischoff B. and Phillips L.D. (1982) Calibration of probabilities: The state of the art to 1980. In Kahneman D., Slovic P. and Tversky A. (eds.) *Judgement Under Uncertainty: Heuristics and Biases*. Cambridge University Press: Cambridge

Long J. and Dowell J. (1989) Conceptions of the discipline of HCI: Craft, Applied Science, and Engineering. In Sutcliffe A. and Macaulay L. (eds.) *People and Computers V*, 9-32. Cambridge University Press: Cambridge

Lu I. and Mantei M. (1991) Idea management in a shared drawing tool. *Proceedings of European Computer Supported Cooperative Work Conference (ECSCW’91)*

Lubars M. (1989) Representing design dependencies in the issue-based information system style. *Technical Report STP-426-889,* Microelectronics and Computer Technology Corporation

MacLean A., Bellotti V. and Young R.M. (1990) What rationale is there in design? In Diaper D., Gilmore D., Cockton G. and Shackel B. (eds.) *Human-Computer Interaction: Proceedings of Interact’90,* 207-212. Elsevier Science Publishers: N. Holland

MacLean A., Shum S. and Bellotti V. (1991) QOC design rationale tutorial and workshop: Collected materials. *Amodeus Working Paper RP6/WP10*. Rank Xerox EuroPARC, July 1991

MacLean A., Young R.M., Bellotti V. and Moran T. (1991) Questions, Options, and Criteria: Elements of design space analysis. *Human-Computer Interaction*, 6 (3&4), 201-250

MacLean A., Young R.M. and Moran T.P. (1989) Design Rationale: The argument behind the artifact. *Proceedings of CHI‘89: Human Factors in Computing Systems.* ACM: New York

Malone T.W. (1985) How do people organise their desks? Implications for the design of office information systems. *ACM Transactions on Office Information Systems*, 1, 99-112

Malone T.W., Yu K. and Lee J. (1988) What good are semi-structured objects? Adding semi-formal structure to hypertext. *Sloan School of Management, Working Paper 3064-89-MS,* Massachusetts Institute of Technology

Marshall C.C., Halasz F.G., Rogers R.A. and Janssen W.C. (1991) Aquanet: A hypertext tool to hold your knowledge in place. *Proceedings of Hypertext'91*. ACM: New York

Manicas P.T. (1966) On Toulmin's contribution to logic and argumentation. *Journal of the American Forensic Association*, 16, 181-186

Marshall C.C. (1987) Exploring representation problems using hypertext. *Proceedings of Hypertext'87*, 253-268. ACM: New York

Mayes T., Draper S.W., McGregor A.M. and Oatley K. (1988) Information flow in a user interface: The effect of experience and context on recall of MacWrite screens. In Jones D.M. and Winder R. (eds.) *People and Computers IV*. Cambridge University Press: Cambridge

McCall R. (1986) Issue-Serve systems: A descriptive theory of design. *Design Methods and Theories*, 20(8), 443-458

McCall R., Bennet P., d'Oronzio P., Ostwald J., Shipman F. and Wallace N. (1990) PHIDIAS: A PHI-based design environment integrating CAD graphics into dynamic hypertext. In Rizk A., Streitz N. and André J. (eds.) *Hypertext: Concepts, Systems and Applications*. Cambridge University Press: Cambridge

McCracken D.L. and Akscyn R. (1984) Experience with the ZOG human-computer interface system. *International Journal of Man-Machine Studies*, 21, 293-310

McCroskey J.C. (1965) Toulmin and the basic course. *Speech Teacher*, 14, 91-100

McDermid J.A. (1991) Safety arguments, software and system reliability. *Proceedings of the International Symposium on Software Reliability Engineering*. IEEE Computer Society: Los Alamitos, CA

McGee V.E. (1986) The OWL: Software support for a model of argumentation. *Behaviour Research Methods, Instruments, and Computing*, 18, 108-117

McKerlie D.L. (1991) Exploring the use of design rationales and scenarios as an approach to communicating human factors expertise. *Masters Thesis*, Department of Computing and Information Science, University of Guelph, Ontario

Minneman S. (1991) The social construction of a technical reality: Empirical studies of group engineering design practice. *Ph.D. Thesis,* Stanford University. Also available as a *Technical Report* (SSL-91-22), Xerox Palo Alto Research Center

Monty M.L. and Moran T.P. (1986) A longitudinal study of authoring using NoteCards. *SIGCHI Bulletin*, 18 (2), 59-60. ACM

Moran T.P. and Carroll J.M. (1992, in preparation) *Design Rationale for Human-Computer Interaction* (tentative title). Lawrence Erlbaum Associates: Hillsdale, NJ

Mostow J. (1985) Toward better models of the design process. *The AI Magazine*, Spring, 44-57

Nelson T.H. (1965) A file structure for the complex, the changing, and the indeterminate. *Proceedings of the ACM National Conference*, 84-100

Nelson T.H. (1981) *Literary Machines*. T.H. Nelson: Swarthmore, Pennsylvania

Neuwirth C.M. and Kaufer D.S. (1989) The role of exernal representations in the writing process: Implications for the design of hypertext-based writing tools. *Proceedings of Hypertext'89*, 319-341. ACM: New York

Neuwirth C.M., Kaufer D.S., Chandok R. and Morris J.H. (1990) Issues in the design of computer support for co-authoring and commenting. *Proceedings CSCW'90: Computer Supported Cooperative Work*, 183-195. ACM: New York

Newell A. and Card S. (1985) The prospects for psychological science in human-computer interaction. *Human-Computer Interaction*, 1 (3), 209-242

Newell A. and Simon H.A. (1972) *Human Problem Solving.* Prentice-Hall: Englewood Cliffs, N.J.

Newman S. and Marshall C. (1990) Pushing Toulmin too far: Learning from an argument representation scheme. *Technical Report* (forthcoming), Xerox Palo Alto Research Center

Norman D.A. (1981) Categorisation of action slips. *Psychological Review*, 88, 1-15

Norman D.A. (1984) Design rules based on the analyses of human error. *Communications of the ACM*, 26 (4), 254-258

Norman D.A. (1986) Cognitive engineering. In Norman D.A. and Draper S.W. (eds.) *User Centered System Design*, 31-61. Lawrence Erlbaum Associates: Hillsdale, New Jersey

Norman D.A.(1988) Distributed cognition. *Unpublished manuscript.* Department of Cognitive Science, University of California, San Diego

Norman D.A. (1991) Cognitive artifacts. In Carroll J.M. (ed.) *Designing Interaction: Psychology at the Human-Computer Interface*. Cambridge University Press: New York

Olson G. (1990) Collaborative work as distributed cognition. *Working Paper*, Cognitive Science and Machine Intelligence Lab, University of Michigan

Olson G.M and Olson J.S. (1991, in press) User centered design of collaboration technology. *Journal of Organisational Computing*

Olson G.M., Olson J.S., Carter M.R. and Storrøsten M. (in press) The activities during design meetings: An example of collaborative work. *Human-Computer Interaction*

Parnas D. and Clements P.C. (1985) A rational design process: How and why to fake it. *IEEE Transactions on Software Engineering*, SE-12, 251-257

Payne S.J. (1991) Display-based action at the user interface. *International Journal of Man-Machine Studies*, 35 (3), 275-427

Petersen D. (1987) Software design capture. *Technical Report STP-138-87*, Microelectronics and Computer Technology Corporation

Potts C. and Bruns G. (1987) Recording the reasons for design decisions. *Technical Report STP-304-87*, Microelectronics and Computer Technology Corporation

Raymond D. (1991) Characterising visual languages. *IEEE Workshop on Visual Languages*, Kobe, Japan

Reason J. (1984) Lapses of attention in everyday life. In Parasuraman R. and Davies D.R. (eds.) *Varieties of Attention*. Academic Press: NY

Rein G.L. and Ellis C.A. (1991) rIBIS: A real-time group hypertext system. *International Journal of Man-Machine Studies*, 24, 349-367. Also in Greenberg S. (ed.) *Computer Supported Cooperative Work and Groupware*, 223-241. (1991) Academic Press: London

Reucker L. and Seering W.P. (1991) A dialectical reasoning system for design documentation. *Proceedings of Third American Society of Mechanical Engineers Conference on Design Automation (Design Theory and Methodology track).* ASME: New York

Rittel H.W.J. (1972) Second generation design methods. (Interview in) *The DMG 5th Anniversary Report: DMG Occasional Paper No.1*, 5-10; reprinted in Cross N. (ed.) *Developments in Design Methodology*, 317-327 (1984). J. Wiley & Sons: Chichester

Rittel H.W.J. and Webber M.M. (1973) Dilemmas in a general theory of planning. *Policy Sciences*, 4, 155-169. Reprinted in Cross N. (ed.) *Developments in Design Methodology*, 135-144 (1984). J. Wiley & Sons: Chichester

Robertson G.G., Mackinlay J.D. and Card S.K. (1991) Cone Trees: Animated 3D visualisations of hierarchial information. *Proceedings of CHI'91: Human Factors in Computing Systems*, 189-194. ACM: New York

Rosson M.B., Maass S. and Kellogg W.A. (1987) Designing for designers: An analysis of design practice in the real world. *Proceedings of CHI and GI'87: Human Factors in Computing Systems and Graphic Interface*, 137-142. ACM: New York

Rosson M.B., Maass S. and Kellogg W.A. (1988) The designer as user: Building requirements for design tools from design practice. *Communications of the ACM*, 31 (11), 1288-1298

Rouse W.B. (1986) A note on the nature of creativity in engineering: Implications for supporting system design. *Information Processing and Management*, 22 (4), 279-285

Russell D.M., Moran T.P. and Jordan D.S. (1987) The Instructional Design Environment. In Psotka J., Massey L.D. and Mutter S.A. (eds.) *Intelligent Tutoring Systems: Lessons Learned*. Lawrence Erlbaum Associates: Hillsdale, New Jersey

Schön D.A. (1983) *The Reflective Practioner: How Professionals Think in Action*. Basic Books: New York

Schön D.A. (1988) Designing: Rules, types and worlds. *Design Studies*, 9 (3), 181-190

Schuler W. and Smith J. (1990) Author's Argumentation Assistant (AAA): A hypertext-based authoring tool for argumentative texts. In Rizk A., Streitz N. and André J. (eds.) *Hypertext: Concepts, Systems and Applications*. Cambridge University Press: Cambridge

Sharples M. and Pemberton L. (1988) Representing writing: An account of the writing process with regard to the writer's external representations. *Cognitive Science Report 119*, School of Cognitive and Computing Science, University of Sussex

Sharples M., Goodlet J.S. and Pemberton L. (1989) Designing the Writer's Assistant. In Williams N. and Holt P. (eds.) *Computers and Writing: Models and Tools.* Blackwell: Oxford

Shneiderman B. and Morariu J. (1986) The Interactive Encyclopaedia System (TIES). *Technical Report*, Department of Computer Science, University of Maryland

Shum S. (1990) Real and virtual spaces: Mapping from spatial cognition to hypertext. *Hypermedia*, 2 (2), 133-158

Shum S. (1991a) Cognitive issues in representing design reasoning as hypertext. *SIGCHI Bulletin*, 23 (1), 38-40. ACM

Shum S. (1991b) Cognitive dimensions of design rationale. In Diaper D. and Hammond N.V. (eds.) *People and Computers VI*, 331-344. Cambridge University Press: Cambridge

Siddiqui J.I.A. (1985) A model of program designer behaviour. In Johnson P. and Cook S. (eds.) *People and Computers: Designing the User Interface*. Cambridge University Press: Cambridge

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

Smith J.C., Weiss S.F. and Ferguson G.J. (1987) A hypertext writing environment and its cognitive basis. *Technical Report TR87-033*, Department of Computer Science, University of North Carolina at Chapel Hill

Smith S.L. and Mosier J.W. (1984) Design guidelines for user-friendly interface software. *Technical Report MTR-9420*, MITRE Corporation: Bedford, Massachusetts

Smolensky P., Fox B., King R. and Lewis C. (1988) Computer-aided reasoned discourse or, How to argue with a computer. In Guindon R. (ed.) *Cognitive Science and its Application to Human-Computer Interaction*, 109-162. Ablex: Norwood, New Jersey

Steele G.L. (1984) *Common LISP: The Language*. Digital Press: Burlington, Massachusetts

Stefik M., Foster G., Bobrow D.G., Kahn K., Lanning S. and Suchman L. (1987) Beyond the chalkboard: computer support for collaboration and problem solving in meetings. *Communications of the ACM*, 30 (1), 32-47

Streitz N., Hanneman J. and Thüring M. (1989) From ideas and arguments to hyperdocuments: Travelling through activity spaces. *Proceedings of Hypertext'89*, 343-364. ACM: New York

Suchman L.A. (1987) *Plans and Situated Actions – The Problem of Human-Machine Communication.* Cambridge Universty Press: Cambridge, UK

Sugiyama K. and Misue K. (1990) 'Good' Graphics Interfaces for 'Good' Idea Organisers. In Diaper D., Gilmore D., Cockton G. and Shackel B. (eds.) *Human-Computer Interaction: Proceedings of Interact'90*, 521-528. Elsevier Science Publishers: N. Holland

Tang J.C. (1989) Listing, drawing and gesturing in design: A study of the use of shared workspaces by design teams. *Technical Report SSL-89-3*, Xerox Palo Alto Research Center

Tang J.C. (1991) Findings from observational studies of collaborative work. *International Journal of Man-Machine Studies*, 34 (2), 143-160. Also in Greenberg S. (ed.) *Computer Supported Cooperative Work and Groupware*, 11-28. (1991) Academic Press: London

Tang J.C. and Minneman S.L. (1990) VideoDraw: A video interface for collaorative drawing. *Proceedings of CHI'90: Human Factors in Computing Systems*, 313-320. ACM: New York

Tang J.C. and Minneman S.L. (1991) VideoWhiteboard: Video shadows to support remote collaboration. *Proceedings of CHI'91: Human Factors in Computing Systems*, 315-322. ACM: New York

Tatar D.G., Foster G. and Bobrow D.G. (1991) Design for conversation: Lessons from Cognoter. *International Journal of Man-Machine Studies*, 34, 185-209. Also in Greenberg S. (ed.) *Computer Supported Cooperative Work and Groupware*, 55-80. (1991) Academic Press: London

The Concise Oxford Dictionary (7th Edition) (1982). Oxford University Press: Oxford

Thomas J.C. and Carroll J.M. (1979) The psychological study of design. *Design Studies*, 1 (1), 5-11

Thomas J., Lyon D., Miller L. and Carroll J.M. (1977) Structured and unstructured aids to problem solving. *Paper presented to American Psychological Association Meeting*, San Francisco

Toulmin S. (1958) *The Uses of Argument*. Cambridge University Press: Cambridge

Treu S., Sanderson D.P., Rozin R. and Sharma R. (1990) Design process and decision rationale models for the N-Chime interface system. *SIGCHI Bulletin*, 22 (1), 73-79. ACM

Ullman D.G.., Stauffer L.A. and Dietterich T.G. (1987) Towards expert CAD. *Computers in Mechanical Engineering*, November-December, 56-70

VanLehn K. (1985) Theory reform caused by an argumentation tool. *Technical Report ISL-11*, Xerox Palo Alto Research Center

Visser W. (1988) Towards modelling the activity of design: An observational study of a specification stage. *Proceedings of IFAC/IFIP/IEA/IFORS Conference on Man-Machine Systems*

Visser, W. (1990) More or less following a plan during design: opportunistic deviations in specification. *International Journal of Man-Machine Studies*, 33(3) 247-278

Walker J.H. (1987) Document Examiner: Delivery interface to hypertext documents. *Proceedings of Hypertext'87*. ACM: New York

Wason P.C. (1966) Reasoning. In Foss B. (ed.) *New Horizons in Psychology*. Penguin: London

Wild C. and Maly K. (1988) Towards a software maintenance support environment. *Technical Report TR-88-04*, Department of Computer Science, Old Dominion University, Norfolk, Virginia

Williams N. and Holt P. (1989) (eds.) *Computers and Writing: Models and Tools*. Blackwell: Oxford

Winograd T. and Flores F. (1986) *Understanding Computers and Cognition: A New Foundation for Design.* Ablex: Norwood, New Jersey

Woodmansee G. (1983) VisiOn's interface design. *Byte*, July, Vol. 7, 166-182

Yakemovic K.C.B. and Conklin J. (1990) Report on a development project use of an issue-based information system, *Proceedings of CSCW'90: Computer Supported Cooperative Work*, 105-118. ACM: New York

Young R.M. (1988) Role of intermediate representations in knowledge elicitation. In Moralee D. S. (ed.) *Research and Development in Expert Systems IV*, 287-288. Cambridge University Press: Cambridge

Appendices

APPENDIX 1: ANALYSIS OF QOC BROWSERS USED IN STUDY 1

'Optimal' browsers to use for Study 1 queries

The table below shows the most efficient use of the available browsers to answer each query.

KEY

q=Question d=Decision o=Option c=Criterion Ct=Criterion tree
H=HyperCard N=NoteCards

(Any of the following browsers would work as graphs or lists)

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
qdH	*qdN*	*qdoN*	*qdcH*	*qdcN*	*qdcN*	*qdocH*	*qdocH*	*qdH +qdN*
10	**11**	**12**	**13**	**14**	**15**			
CtH +CtN +qdcH +qdcN	*qdcN + qdcH*	*qdcN + qdcH*	*CtN + CtH*	*CtN + CtH + qdocN + qodcH*	*CtN + qdcN*			

Query scheduling and use of browsers in Study 1

This log gives information on new, open, and used browsers for each query.

KEY

q=Question d=Decision o=Option c=Criterion Ct=Criterion tree
H=HyperCard N=NoteCards

1 = query number
italics = used by subject for this query
normal text = graphical browser open on desk but not used
<u>underlined text</u> = list browser open on desk but not used
bold text = erroneous use of browser (in terms of whether it provided relevant or sufficient information)
... + = subject spent some time thinking before requesting another browser
Esel: = experimenter selected query, due to lack of time. These queries were usually more complex ones which subject had not reached, but for which data was of particular interest.

Thus:

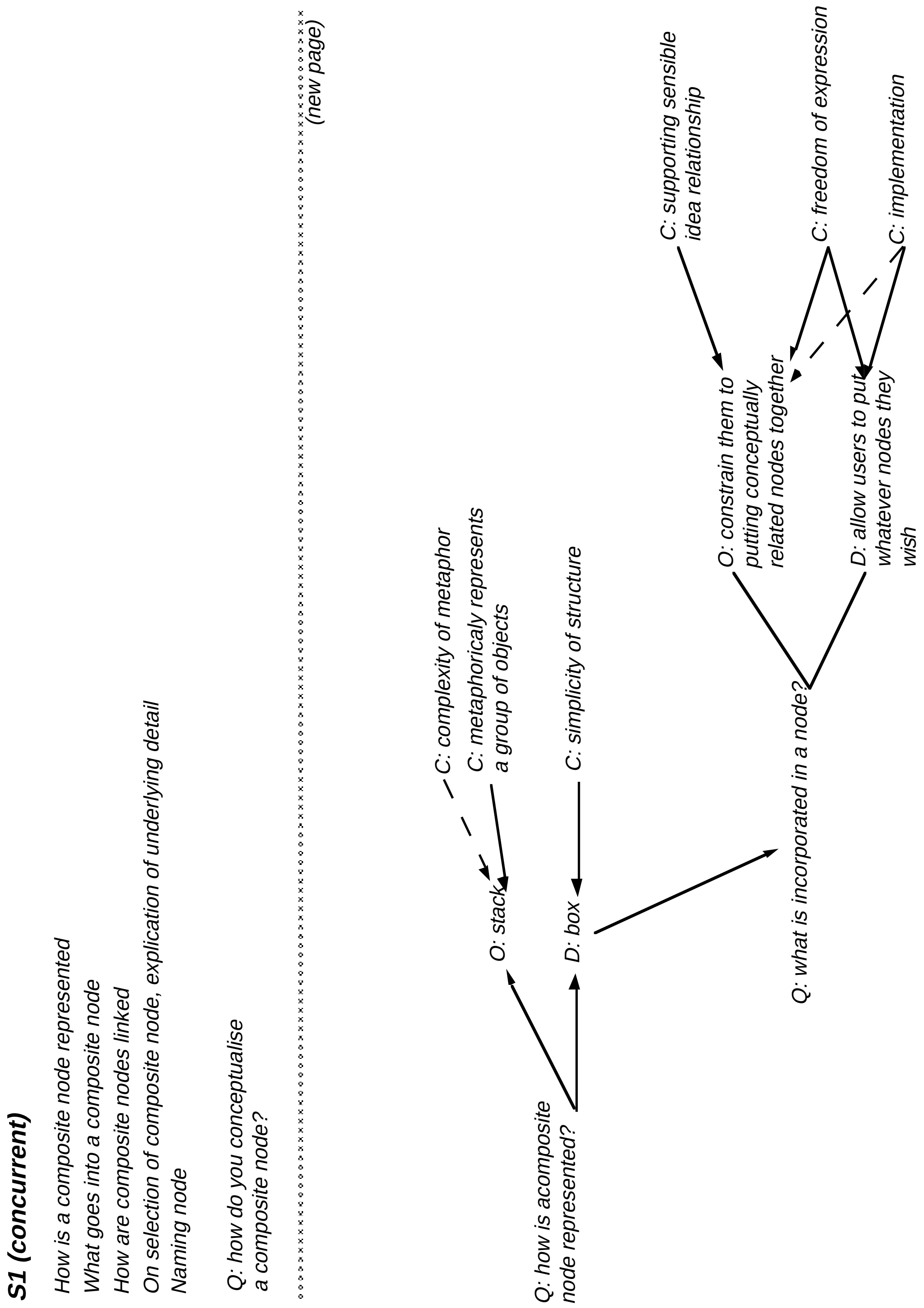
qdN = subject used graph with Questions and Decisions on Notecards.
<u>qdoH</u> = subject used list with Questions, Decisions and Options on HyperCard.
CtN = NoteCards Criterion tree was open on desk, but not used for this query
qdcH + *<u>CtH</u>* = graph for Hypercard, with Questions, Decisions and Criteria was requested in error, and the list version of the HyperCard Criterion tree was used

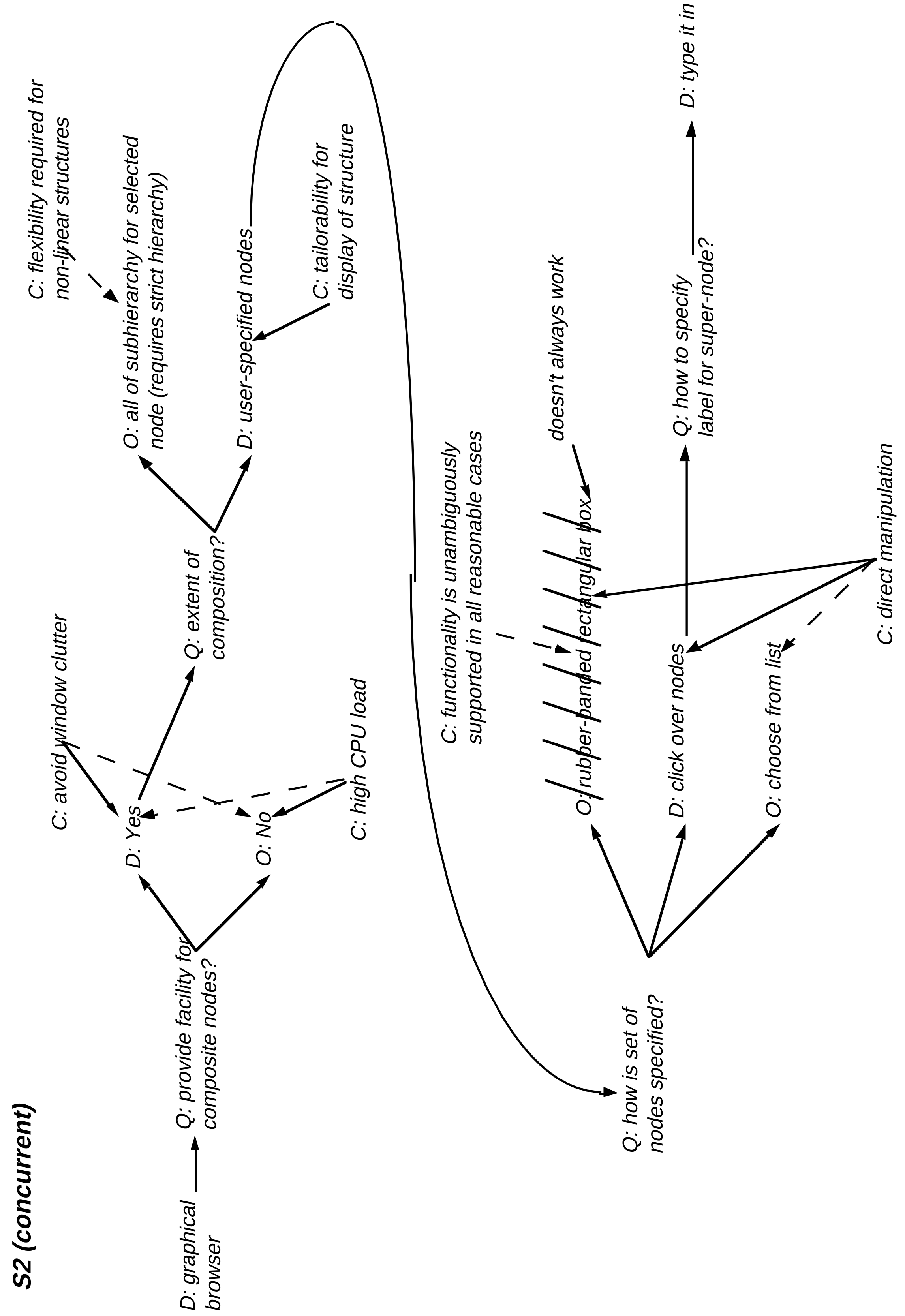
S1	**1** *qdoH*	**4** *qdoH*	**7** ***qdoH*** *qdocH*	**8** *qdoH* *qdocH*	**2** *qdoH* *qdocH* *qdoN*	**3** *qdoH* *qdocH* *qdoN*	**5** *qdoH* *qdocH* *qdoN* *qdocN* *+**CtN***	**6** *qdoH* *qdocH* *qdoN* *qdocN* *CtN*	**15** *qdoH* *qdocH* *qdoN* *CtN* *+qdoc* *N*
	9 *CtN* *qdocN* *+qdoc* *H*	**10** *CtH* *+CtN* *qdocN* *qdocH*	**Esel:1** **4***CtH* *+CtN* *qdocN* *qdocH*	*END*					
S2	**1** *qdH*	**2** *qdN*	**3** *qdoN*	**4** *qdoH* *(cf* **qdoN** *qdoH)*	**5** *qdcN*	**6** *qdcN*	**7** *qdcN* *qdocH*	**8** *qdcN* *qdocH*	**9** *qdH* *+qdN*
	10 *CtH* *+CtN…* *+qdcH* *+qdcN*	**Esel:1** **5** ***CtH...*** *CtN+* *CtH* *(cf)* *+qdcN*	*END*						
S3 *(lists)*	*1qdocH*	**2** qdocH *qdocN*	**2** qdocH *qdcN*	**3** qdocH ***qdcN*** *qdocN*	**4** *qdocH* qdcN qdocN	**5** qdocH qdcN *qdocN*	**6** qdocH *qdocN*	**7** *qdocH* qdocN	**8** *qdocH* qdocN
	9 *qdocH* *+qdoc* *N*	**10** *qdocH* *+qdoc* *N*	**11** *qdocH* *+qdoc* *N*	**12** *qdocH* *+qdoc* *N*	**13** *qdocH* *+qdoc* *N* *…CtH* *+CtN*	**10** qdocH +qdoc *CtH* *+CtN*	*END*		
S4	**1** *qd*	**2** qdH *qdN*	**3** qdH qdN ***qcN*** *qdoN*	**4** qdH qdN qdoN *qdoH*	**5** qdH qdN qdoN qdoH *qdocN* *qdocH*	**6** qdH qdN qdoN qdoH *qdocN* *qdocH*	**7** qdH qdN qdoN qdoH *qdocN* *qdocH*	**8** qdH qdN qdoN qdoH *qdocN* *qdocH*	**9** qdH qdN qdoN qdoH *qdocN* *qdocH*
	10 qdH qdN qdoH *qdocN* *qdocH* *…**qdoN*** ***qdcN*** ***qdcH***	**11** qdH qdN qdoH qdoN qdcN qdcH *qdocN* *qdocH*	**12** qdH qdN qdoH qdoN qdcN qdcH *qdocN* *qdocH*	**13** qdH qdN qdoH qdoN qdcN qdcH *qdocN* *qdocH*	**14** qdH qdN qdoH qdoN qdcN qdcH *qdocN* *qdocH* *(&* *note*	**15** qdH qdN qdoH qdoN qdcN qdcH *qdocH* *qdocN* *(Esel* *CtN)*	*END*		

S5 (queries on cards)	**3** **qdN** qdoN	**5** qdN qdoN qdocN	**3** qdN qdoN qdocN	**6** qdN qdoN qdocN	**15** qdoN qdocN …qdN (pstpnd	**2** qdN qdoN qdocN	**15** qdoN qdN CtN… +qdocN ..+qdcN	**7** **qdcH** qdocH	**8** qdocH
	4 qdocH	**1** qdocH	**9** qdocH qdocN	**11** qdoN qdoH	**14** qdoN qdoH CtH +CtN … qdocH +qdocN	END			
S6 (queries on cards)	**3 doN** qdoN	**15**CtN qdocN	**15** CtN qdocN	**5** CtN qdocN	**2** **CtN** qdocN	**1** CtN qdocN qdocH	**7** CtN qdocN qdocH	**8** CtN qdocN qdocH	**4** CtN qdocN qdocH
	qdocH qdocN CtH CtN	**9** CtH CtN qdocH qdocN	**Esel:13** CtH CtN qdocH qdocN	**Esel:14** CtH CtN qdocH qdocN	END				
S7 (queries on cards)	**1** qdocH	**8** qdocH	**4** qdocH	**7** qdocH	**5** qdocH qdocN	**2** qdocH qdocN	**3** qdocH qdocN	**6** qdocH qdocN	**15** qdocH qdocN +CtN.. .+qdcN
	11 qdcN qdocN + qdocH. .+qdH +qdN	**14** qdcN qdocN qdcN qdN +qdH +qdocH	**13** qdcN qdocH qdocN qdH +qdN	**9** qdcN qdocH qdocN qdH +qdN	**10** qdcN qdocH qdocN qdH qdN CtH +CtN	END			

APPENDIX 2: QOC REPRESENTATIONS FROM STUDY 2

The QOCs from Study 2 have been reproduced, preserving as far as possible the original spatial layout. Subjects who used QOC from the beginning as they tackled the problem are labelled *concurrent*. Subjects who used QOC after they had engaged in initial design for 30 minutes are labelled r*etrospective*—see Study 2 for details [§conditions].




S2 (concurrent)
D: graphical browser
Q: provide facility for composite nodes?
D: Yes
O: No
C: avoid window clutter
C: high CPU load
Q: extent of composition?
O: all of subhierarchy for selected node (requires strict hierarchy)
D: user-specified nodes
C: flexibility required for non-linear structures
C: tailorability for display of structure
Q: how is set of nodes specified?
O: rubber-banded rectangular box
D: click over nodes
O: choose from list
C: functionality is unambiguously supported in all reasonable cases
doesn't always work
C: direct manipulation
Q: how to specify label for super-node?
D: type it in

S3 (concurrent & Lists only)

Decision List:

Aim: produce decision list for "Create Composite Node"
Q1. How to specify which part of the graph to package?
Q2. How to display packaged section?, eg. text?
Q3. How to unfold package?
Q4. Where unfolded section is displayed and how?

(new page)
Q1: Specify package in graph?
O1 : form filling
O2: direct manipulation - drawing around area
D: draw around area

+C: direct manipulation
–C: isn't consistent with interaction style

(new page)
Q2: how to display packaged section?
O1: small box
O2: iconise it
O3:

D: small box

C+: graph easier to read because small
C–: inconsistent with interaction style because difficult to iconise

(new page)
Q: how do you identify your package?
O1: index
O2: text title free to enter
O3: text title inherited from graph

C–: index - not consistent - because existing system uses text only for buttons
C–: O2 - inconsistent because no free text entry for buttons (locus of control restrictions)

D: O3

Q3: unfolding package?

O1: mouse control unfolding
O2: menu option
O3: combination of two

C–: O1 - accidental unfolding
C–: O2 - inconsistent because no mouse control
C+: O3 - consistent with current system

D: O3

(new page)
Q4: where to display unfolded section?

O1: to replace original graph
O2: unpacked away from graph

C–: O2 - confusing because the view would be different to original
C+: O1 - consistent with original view

D: O1

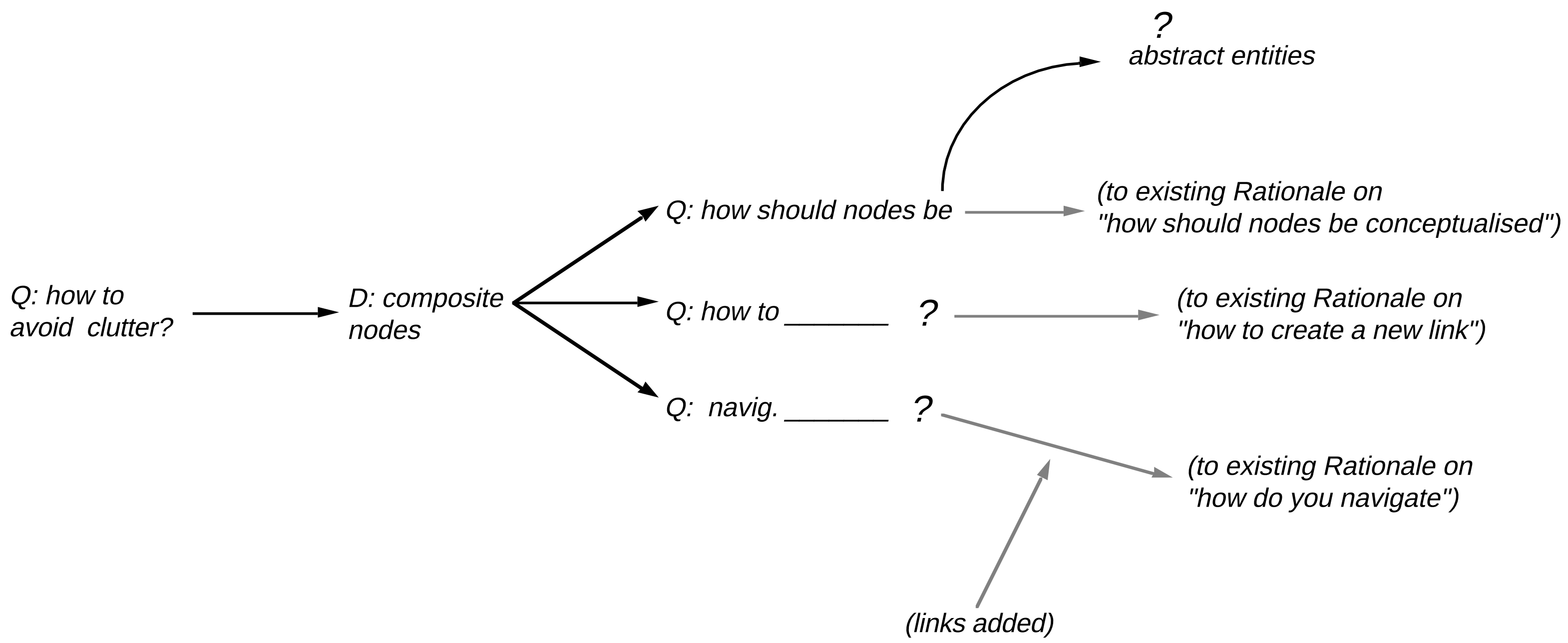
Q5: how to display what's unfolded?

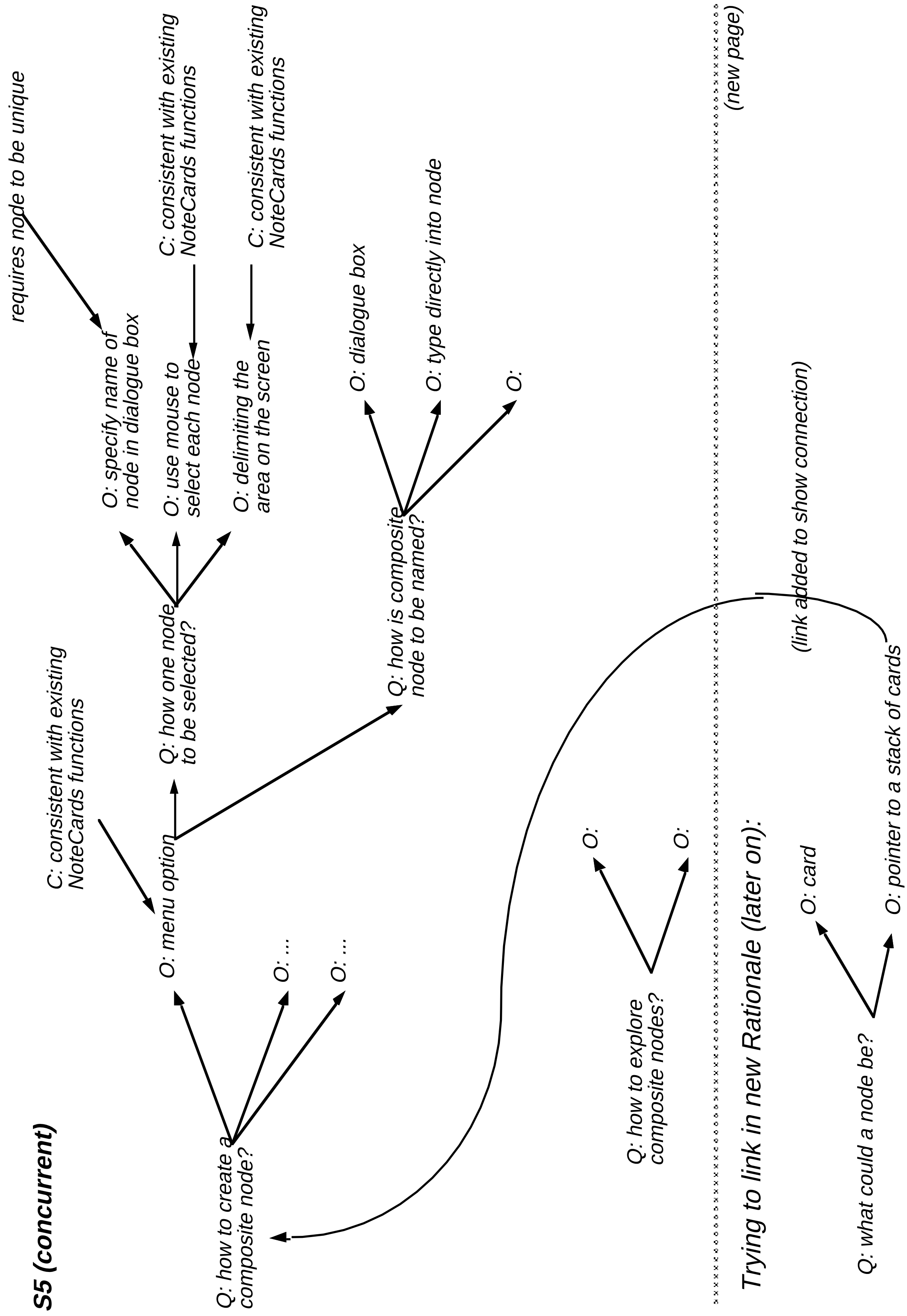
O1: exactly replicate the graph, ie. same colour, same font etc.
O2: do you distinguish what's unfolded

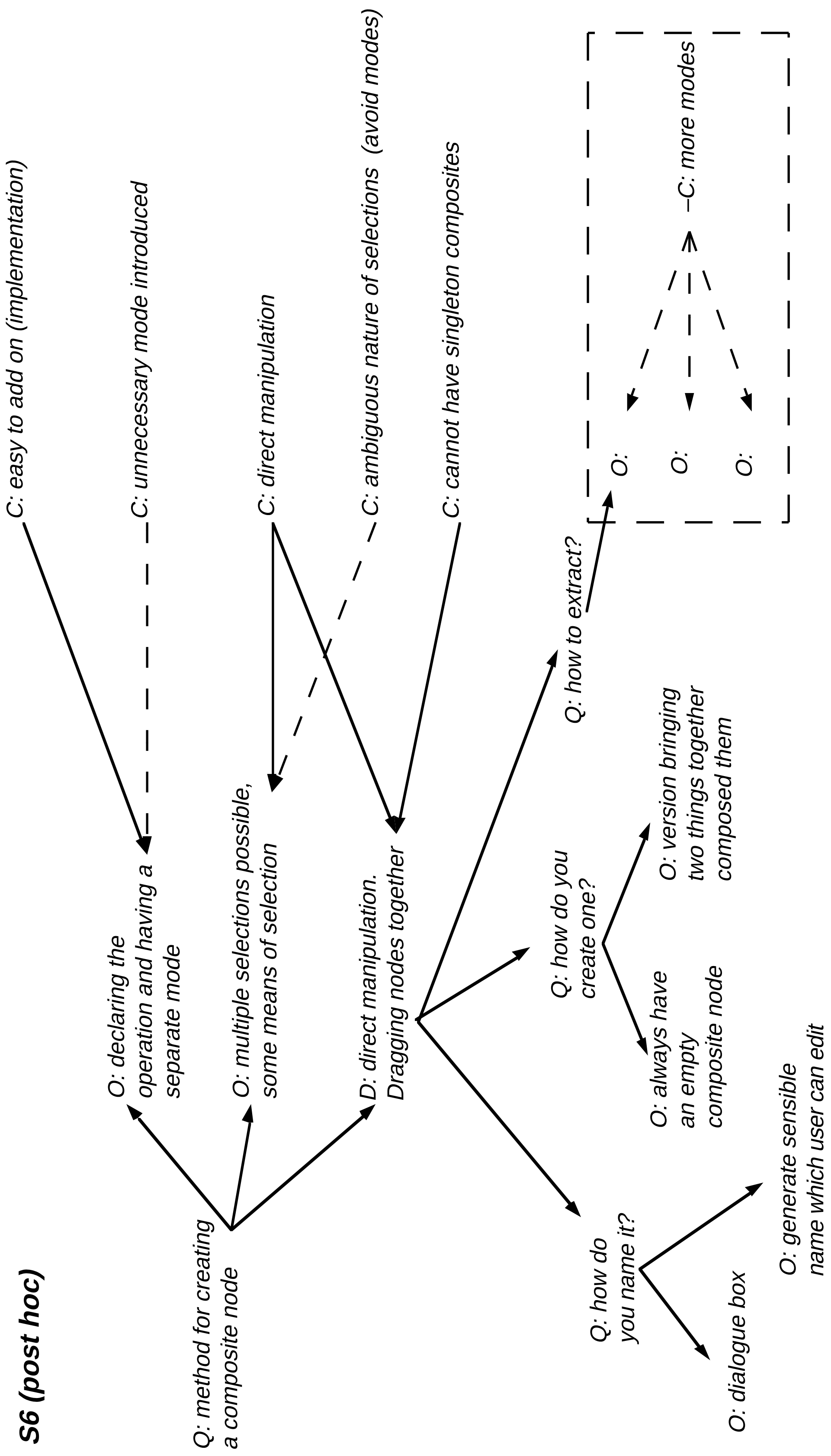
C–: O1 - may forget that it was a package, and redo operation
C+: O2 - distinguish package as a package. Therefore identifying what in the graph had been packaged. Supports partial recall

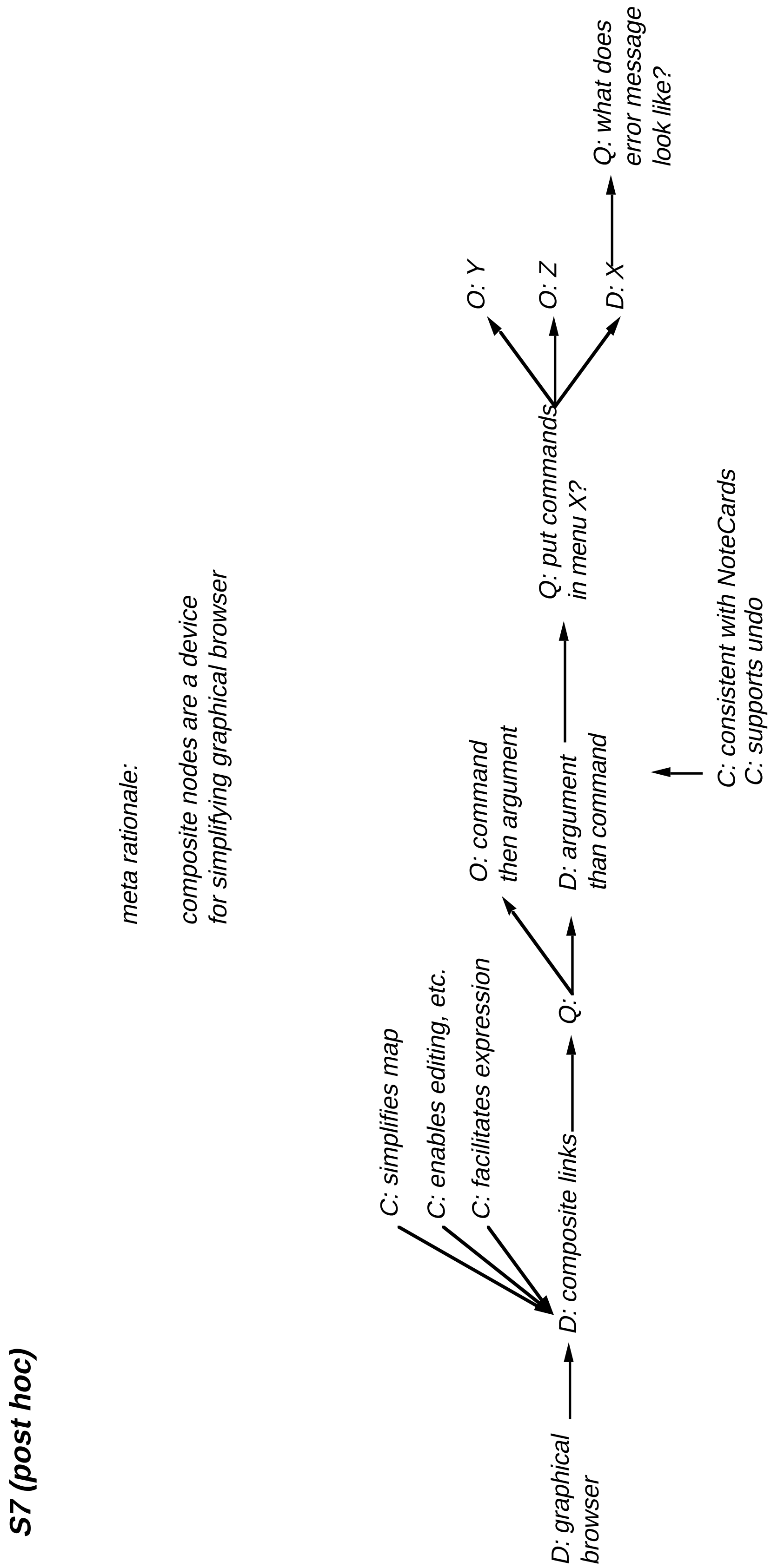
S4 (concurrent)

(Explanatory notes have been added to this QOC based on the designer's explanation of what he wanted to do)




S5 (concurrent)
Q: how to create a composite node?
O: menu option
O: ...
O: ...
C: consistent with existing NoteCards functions
Q: how one node to be selected?
O: specify name of node in dialogue box
requires node to be unique
O: use mouse to select each node
C: consistent with existing NoteCards functions
O: delimiting the area on the screen
C: consistent with existing NoteCards functions
Q: how is composite node to be named?
O: dialogue box
O: type directly into node
O:
Q: how to explore composite nodes?
O:
O:
(new page)
Trying to link in new Rationale (later on):
Q: what could a node be?
O: card
O: pointer to a stack of cards
(link added to show connection)


S6 (post hoc)
Q: method for creating a composite node
O: declaring the operation and having a separate mode
O: multiple selections possible, some means of selection
D: direct manipulation. Dragging nodes together
C: easy to add on (implementation)
C: unnecessary mode introduced
C: direct manipulation
C: ambiguous nature of selections (avoid modes)
C: cannot have singleton composites
Q: how to extract?
O:
O:
O:
C: more modes
Q: how do you create one?
O: version bringing two things together composed them
O: always have an empty composite node
Q: how do you name it?
O: dialogue box
O: generate sensible name which user can edit

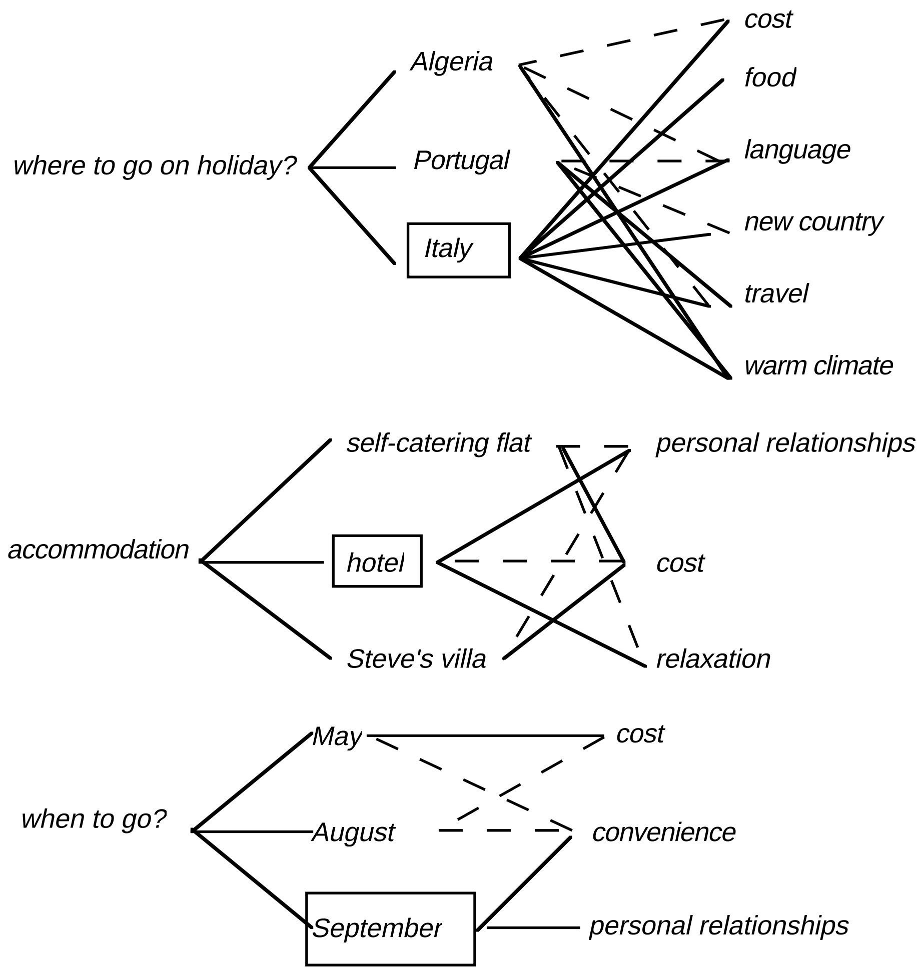

S7 (post hoc)
meta rationale:
composite nodes are a device
for simplifying graphical browser
D: graphical
browser
D: composite links
C: simplifies map
C: enables editing, etc.
C: facilitates expression
Q:
O: command
then argument
D: argument
than command
C: consistent with NoteCards
C: supports undo
Q: put commands
in menu X?
O: Y
O: Z
D: X
Q: what does
error message
look like?

APPENDIX 3: EXAMPLE AND SUMMARY OF QOC, PLUS HINTS ON ITS USE [STUDY 2]

(Note: There were several other brief examples of QOC before this one).

Example: Planning a holiday

Jon	ok, so where're we going this year?
Ann	somewhere *warm*
Tim	don't forget I'm nearly broke
Fay	how about Algeria? - it's hot there!
Jon	perfect, except they don't speak English, and the travelling's terrible!
Ann	yeah I'd prefer not too much travelling - it wastes a day recovering once you're there. Algeria I hear is expensive
Fay	ok ok! ... well we'd get a tan in Portugal, and it's not far to go
Ann	sorry, I've been there - I want to see *new* places
Tim	how about Italy? Good exchange rate, pretty close, and good food
Ann	that's one I haven't been to
Jon	we could stay really cheaply in my friend's villa, though he'll be there as well
Fay	is that Steve? I can't stand him - I'd prefer a hotel
Tim	or self-catering somewhere else? – bit cheaper. Don't forget we all did a year's Italian—4 year's ago! They all speak English out there anyhow.
Fay	well I'd really like to be waited on, and have a break from cooking and washing-up - it always gets left to the women anyhow. I'm prepared to splash out for a cheap hotel.
Jon	well, *I'm* certainly not slaving for you - it's womens' work anyhow isn't it?!
Ann	er...to keep the peace during the holiday, perhaps it would be better if we go for a hotel? I'm not that keen on Steve either and I've just had a raise - is the money side ok with you Tim? By the way, when are we going?
Tim	well ok - I'll break into my piggy bank savings then. How's August with everyone?
Jon	I've got all summer free – and cheap hotel sounds fine to me
Fay	I think May or September to avoid the worst crowds and peak prices
Tim	May's out I'm afraid - work's terrible then
Ann	I'm in Germany end of May, but September's fine
Fay	yep, that's ok with me - Mike's offered to take me to Bognor Regis then, so this'll be a good excuse to get out of it!


cost
food
Algeria
language
where to go on holiday?
Portugal
new country
Italy
travel
warm climate
self-catering flat
personal relationships
accommodation
hotel
cost
Steve's villa
relaxation
May
cost
when to go?
August
convenience
September
personal relationships

Note the following conventions:

• **put a box round the Option which is chosen** to show it is a decision

• use **solid lines** to show *supports* relationships from a Criteria to Options, and **dashed lines** to show *objects-to* relationships.

• so that a solid line always indicates a 'plus' for an Option, and a dashed line a 'minus', **Criteria are positive**, eg. *simplicity* rather than *complexity*; *ease of use* rather than *difficulty of use*. Neutrally expressed Criteria like *learnability, cost, image*, *weight* etc. can be used sometimes, as it's clear what their meaning is if they object to or support an Option.

• you can see that **lots of new Issues** (and subsequent Options and Criteria) are raised following the decision to go electronic. Thus, the final record of any discussion will have many Issues.

General hints:

• **a link should only be made if the Criterion directly affects the Option**. It is not necessary to try and decide on a link from every Criterion to every Option if it doesn't make sense. In the Holiday example, the Criterion *personal relationships* was only relevant to September, so links to the other two Options are left out.

• **don't be too vague in naming Criteria.** An extreme example of overgenerality would be to use a Criterion like *good* to summarise everything. Use Criteria which make clear what it is about an Option which distinguishes it from the others.

Having said that...

• **Criteria are often abstracted** from the discussion and expressed in a more general way, eg. choosing what month to go in the Holiday example, the Criterion *convenience* covers reasons to do with crowds, prices, and availability from work – but if you thought it was necessary, you could represent these as three separate Criteria; another example is that nobody actually said "We have a *personal relationships* problem here" – but that is a valid way to pull together reasons to do with the personality clashes which affected a couple of decisions.

• **Criteria can be reused** (eg. cost; software speed). Normally, as the design takes shape, various Criteria emerge as particularly important ones, and influence subsequent decisions. Thus, *learnability* or *familiarity* might be recurring factors when designing the way information is entered, located, and deleted.

• **the notation summarises arguments over the whole session**, ie. you are not limited to representing things in the order they were raised during discussion. For example, if the Issue of *where to go on holiday* had arisen later on in the discussion, you would go back to the rationale created earlier and add to that, rather than start another Issue: *where to go on holiday?*

Summary

Generating design rationale should (i) be helpful to you as designers, and (ii) convey to others the reasoning behind your final design. More specifically:

(i) organising your ideas using this notation should help you think more thoroughly as to why exactly you make the decisions you do;

(ii) design rationale should be organised so that an outsider can understand the issues underlying a design, the possibilities each offers, and reasons why one rather than another was chosen. Coherent rationale *may* emerge by just writing down Issues, Options and Criteria as you think of them, but not necessarily – it is often necessary to reflect on the best way to organise and name these, and if necessary change the notation first put down put down to make these factors clearer.

The final rationale is an idealised account of how you tackled the problem – like a textbook example – rather than a literal record of the misunderstandings, false starts etc. that were made.

APPENDIX 4: QOC TRAINING EXERCISE 1 – SHORT DESIGN EXERCISES [STUDY 3]

Notational Practice

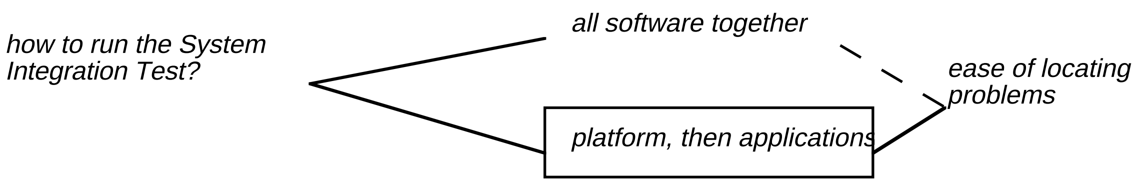
Here are 10 brief extracts of design discussion in a variety of areas.

Try to represent the reasoning in the arguments as notational Issues, Options, and Criteria. It is up to you to decide how many Issues there are in each example. Note that in some cases a decision is not always made.

Below are two worked examples:

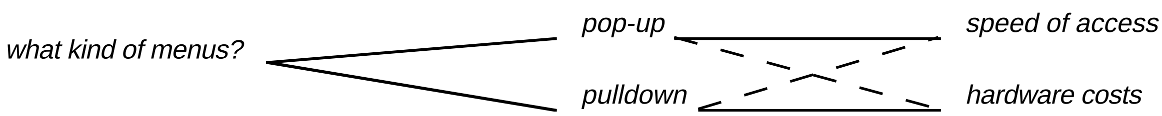
Example 1

- So, how are we going to perform the System Integration Test?
- well, we could just run all the software together and see what happens...
- that makes it hard to figure out where problems are though. Let's do the platform software first and then the application software
- ok, so then the application will have a solid base for testing.

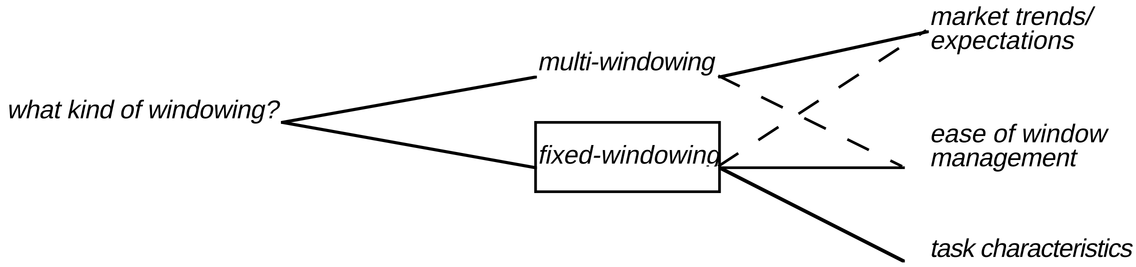


Example 2

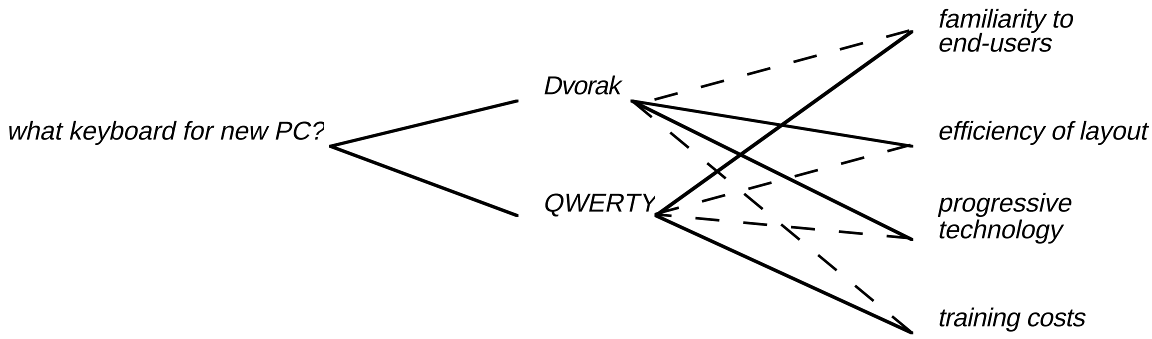
- pop-up menus are probably the best thing to go for - click anywhere on the screen with the righthand button, and it appears instantly at that point. No need to move the mouse up to the top of the screen everytime.
- the problem is with the hardware guys - they've said they want to try a cheaper one-button mouse – we may be forced to go for pull-down menus at the top of the screen.



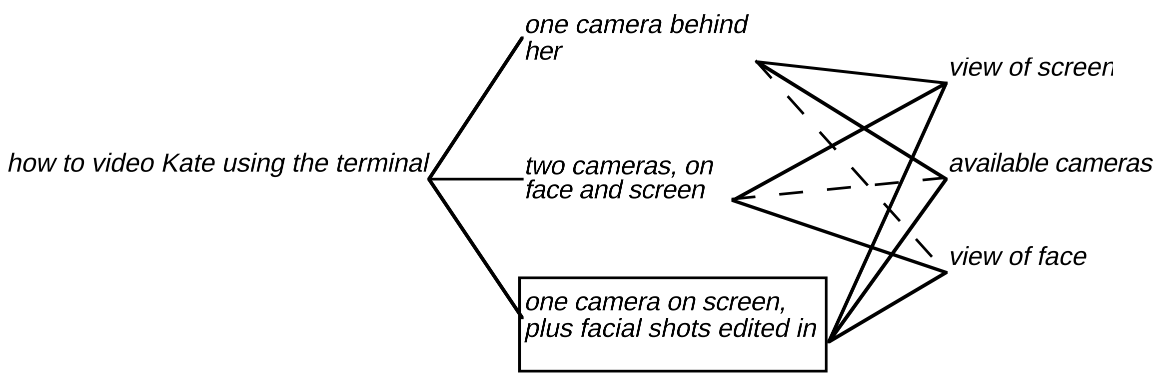
1
- we've got to go for a multiple overlapping window display – the market demands it now in all new applications
- yes, but we're trying to produce the most *usable* software we can. Multi-windowing can cause serious problems with losing windows and continual resizing. Four or five windows fixed in size and position may serve our purposes best. For this application, all our users are going to do is jump between those five different displays.
- hmm, alright.



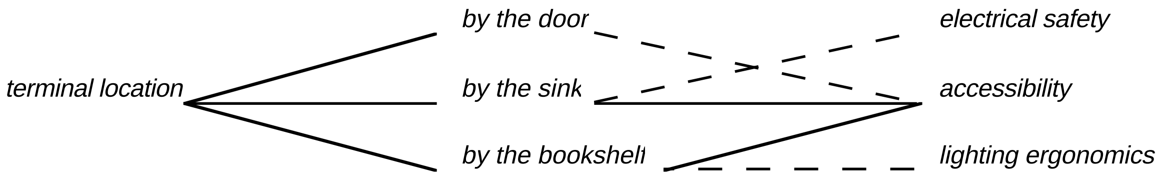
2
- as a forward looking company, I really think we should introduce the Dvorak keyboard as the standard for our new PC.
- you've got to be joking. Nobody knows how to use it – the whole world uses QWERTY keyboards.
- the QWERTY is a hangover from old technology. It is extremely inefficient in its layout - it's our duty to support the Dvorak introduction - other companies are bringing it in
- but nobody would buy it! All our customers would immediately demand normal keyboards, or go elsewhere. By the time they've got all their staff trained up on it, any supposed benefits from its layout will be lost



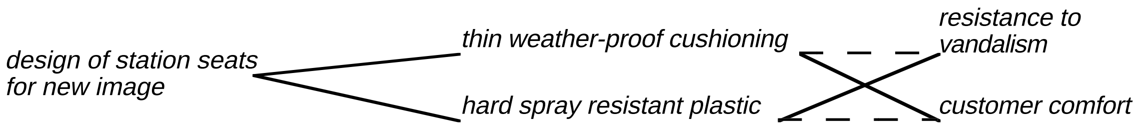
3
- Right, I think the best way to video Kate at the terminal is from behind and to the side, so we get her head in, and still see the screen.
- but Steve wants to get in some facial expression, or 20 minutes of screen activity gets boring. We need a full facial shot to capture that properly – I think we want two cameras for face and screen, and then we mix the two images later
- but our second camera got bust last week. We're going to have to shoot the screen first, and then record bits of facial expression afterwards, and edit them in at appropriate points.



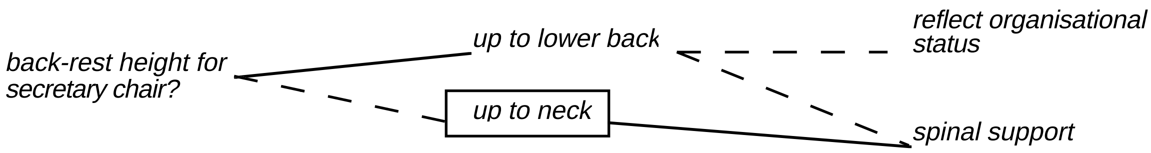
4
- right then, where's the terminal for the mainframe going to go? Over there by the door?
- no that's no good – you'd have to sit in the doorway to use it
- I thought over by the sink, out of the way
- ...where it'll be splashed and short-circuited within a week
- facing the bookshelf over on the right?
- there's too much light from the window glaring on the screen, and it's in the way of anyone wanting to get at the middle bookstack

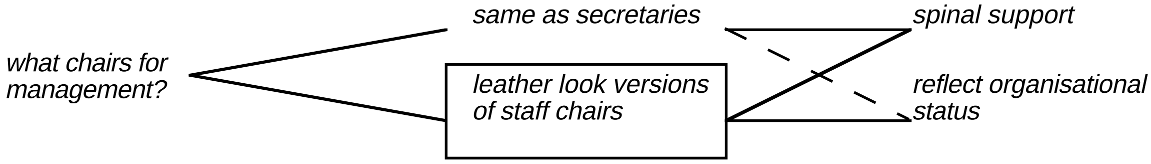


5
- this seat is going to be for waiting rail commuters, on unmanned suburban platforms. Does anyone think we should provide cushioning?
- Network Dumberside's new image of high quality service suggests to me that we should keep our clients happy, and provide some thin, weatherproof padding.
- Oh come on – we all know it'll be slashed or sprayed within a week of introduction—then what will our image be? We can do perfectly respectable seats in hard, spray resistant plastic. I'm sure our customers will appreciate unvandalised seats as opposed to padded ones in ribbons and sprayed, even if they aren't like their living-room sofas

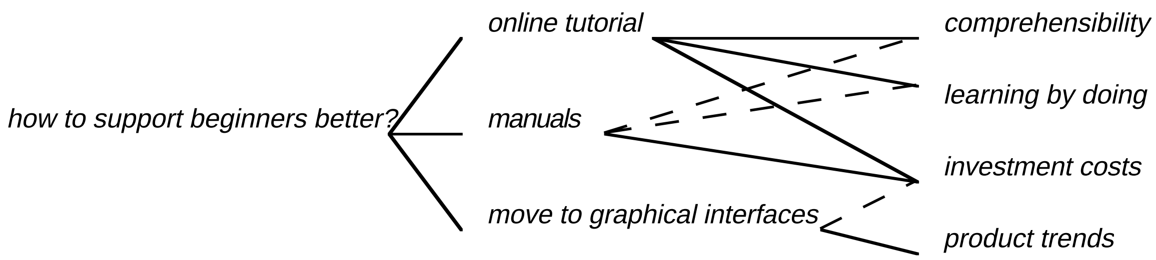


6
- our new chair is targetted at secretaries doing wordprocessing work. Can we come to a decision about the back-rest design?
- a low back-support should be quite adequate, up to the small of the spine.
- no, that's no good when you're sat all day at a screen. The human spine requires full support up to the base of the neck.
- but what are the managers going to say when their junior staff take charge of chairs which look like theirs? - it's a big status thing you know.
- well they've just got to realise that their staff are humans too – everyone requires anatomically sound chairs.
- we could still preserve the difference by developing a flashier leather look one, but basically the same thing underneath.
- yes, that sounds like the best of both worlds

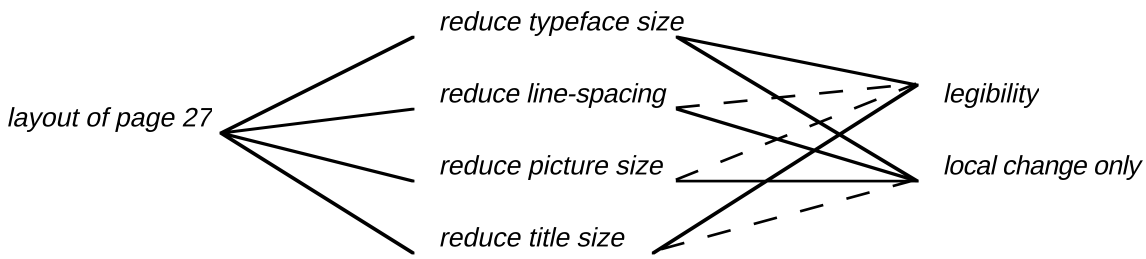




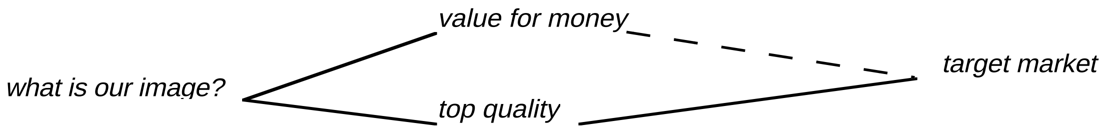
7
- ok, as you all know we're looking to provide more help to beginners with our software – ideas?
- the real problem's with our manuals – far too complex. The technical writers really need to get their act together, as the user is put off immediately if the manual's bad.
- I think we really need a tutorial on a floppy disk, which takes beginners through the first steps by allowing them to try parts of the software in a controlled situation, with help on-hand.
- no major changes to the actual software itself required there
- No, no: basically, we need to make the move from text oriented software to graphical mouse and icon based stuff – beginners find that so much more natural.
- uh-huh, but we're talking about a major investment in time and money there.
- but that's the way things are moving in the industry, and the sooner we get a mouse/menu based Version 3.1 out, the better.

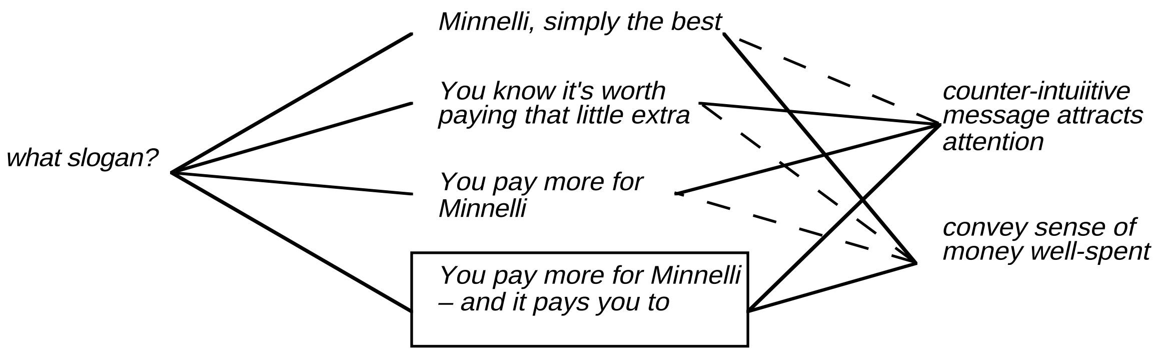


8
- somehow, we've got to fit these two paragraphs and 3 pictures into page 27.
- one way is to decrease typeface size, but we can't get away with reducing the spacing between lines anymore
- can the pictures be reduced some more?
- I think we'll lose too much detail with the quality of printer we have.
- why not reduce the size of the title? It's taking up a lot of space
- if we do, we'll have to reduce all the other titles as well to be consistent, and that'll change the layout of the preceding pages. Try and avoid global changes like that if we can.

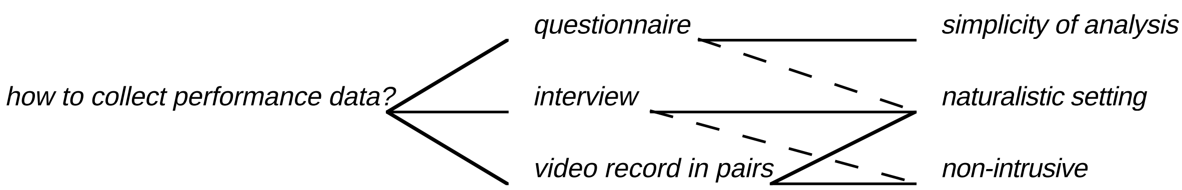


9
- OK, we're looking for a slogan to really convey Minnelli quality
- "Minnelli – simply the best"
- come on, that's about as boring as you can get.
- do we want to talk about value for money?
- no, we're targetting the top end of the market here - budget-buy slogans aren't really appropriate
- well, how about, "You know it's worth paying that little extra"?
- hmmm, so this is playing on the idea that you make a point of saying how expensive your product is...
- yeah, so the reader is surprised and reads on
- well in that case, let's do it properly with something like, "You pay more for Minnelli"
- that's a bit blunt - what does the customer get in return? We'd need to add something along the lines of, "You pay more for Minnelli – and it pays you to".





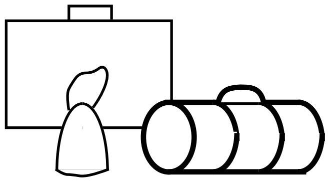
10
- somehow, we've got to find out what problems the secretaries have using our wordprocessor.
- that's fine – the simplest way is to give them the User Satisfaction Questionnaire. All we have to do is score them.
- I have my doubts about that. The questions aren't realistic, and people can't always remember or even articulate all the difficulties they've found. I feel we should be measuring them as they actually use the system rather than afterwards
- why not ask them questions then? Someone can sit in during their practice sessions and ask them to explain what's going on
- that's intrusive – they need to be able to work as though they're alone in their office
- well this sounds like a good case for video recording. If we put them in pairs, and ask them to think aloud, we should get a reasonable feel for their experience of the software.



APPENDIX 5: QOC TRAINING EXERCISE 2 – GRAPHIC DESIGN DIALOGUE [STUDY 3]

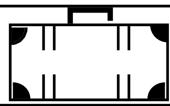
Designing a public information sign for an international airport, to mean: "1 hour left-luggage office". (No text allowed)

Mike This is a tricky sign ... well the luggage bit is easy – just draw a few bags



Neil but you've got to be able to make out what they are, and if there're lots of little bags, the whole thing gets visually cluttered.

Mike a single case which can be easily recognised is better, standing for all kinds of baggage



Neil well ok, just one case then ... do you think we should have any people in it?

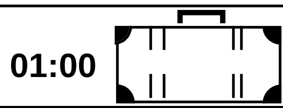
Mike well, we've got to communicate that the luggage is 'left there' somehow – we're not selling bags, or a lost luggage office. You could have a queue of people waiting to reclaim... no that's a bit obscure

---

Neil not quite relevant

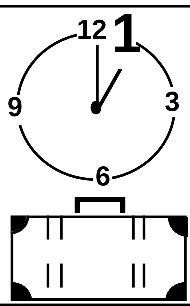
Mike something to do with time? - well, let's be fairly literalistic and have an hourglass...

Neil we could be up to date and have a digital clock instead showing 01:00! - something like this



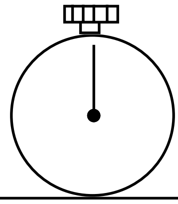
Mike do you think people'll get that? Isn't a standard clock with hands and numbers the most obvious thing? I mean you could get folk from places where they don't have digital watches - but analogue dial watches are still very much around aren't they?

Neil ok - and we could highlight the 'one', and stick it above the suitcase.



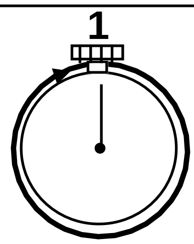
Mike The crucial thing is to only have the basic elements - no extra information which could lead folk astray. We're trying to show *one hour passing* ... but it seems to say 'one o'clock' at the moment...

Neil have a stopwatch? Just take all the numbers out, and remove one of the hands and add the start/stop button at the top



Mike ...a stopwatch which has gone a full hour - how do we show that?

Neil well put an arrow round it, like this... and stick a ‘1’ at the top?

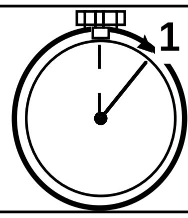


Mike I think everyone’ll know that sort of ‘1’ - I mean we *could* put in a whole bunch of ‘1s’ in different languages, but I think that’s going over the top - and it’d get really cluttered as well

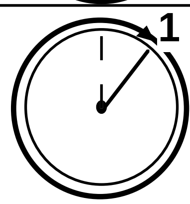
Neil nah, that’s ok ... but the hand looks like it’s pointing at twelve now!

Mike oh. yeah - it should show some change - we’re back to trying to show one hour passing...

Neil ok, what about a dotted line pointing up, and a solid line pointing to about the ‘one’ position, and put the ‘one’ here rather than at the top

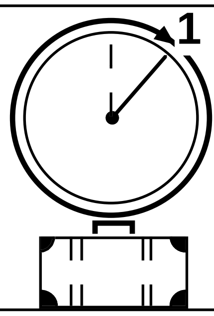


Mike it’s back to looking like a clock! - well, a simplified one - but clarity’s what we’re after. And it just occurred to me that some foreigners may not be familiar with the shape of a stopwatch – or an hour-glass for that matter! So perhaps if we take off the start/stop button at the top it’ll look right...



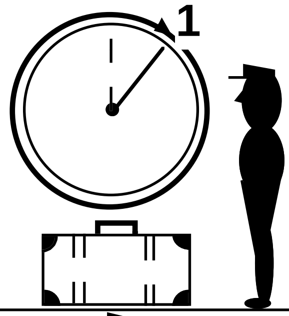
Neil yeah! That’s good - you’d really have to be dense not to get the message now

Mike well, let’s put the clock and case together to see the total effect - hmmm, not bad.

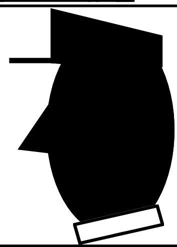


Neil I think we should have a person in there, ... someone official looking to show that the case is being looked after - it’s *safe* - rather than just sitting there unattended.

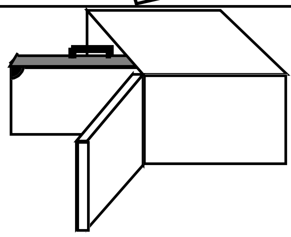
Mike how about a bloke in a cap and uniform next to the case, something like this ... a baggage attendant chappie



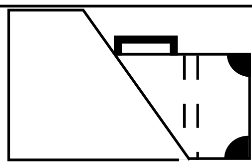
Neil yeah, but he's too small - you might not figure out what he's supposed to be from a distance - make him bigger - no: just have the head and neck - how's that?



Mike pretty good - and if we put him on the right, it looks like he's watching the case! I guess the only question I'm asking now is, "What's this guy supposed to be doing with the bag?" - you know he could be a cop who's found it - if we really want to show the luggage is alright, we should put it in a safe or something

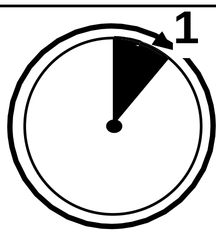


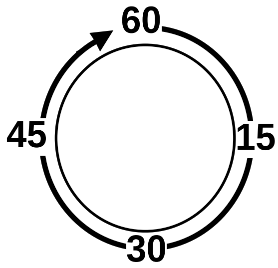
Neil the idea's good, but the three dimensional drawing's getting really complicated, though it looks more realistic. It's like having four bags instead of one - just keep it kinda abstract and two-dimensional. Otherwise we'd have to add depth to everything to keep it consistent



Mike yep, that's pretty good ... but looking at it, the hands still make it look too much like 'one o'clock' - it could mean 'collect luggage after one o'clock'. We've got to avoid anything which could be read as a specific time

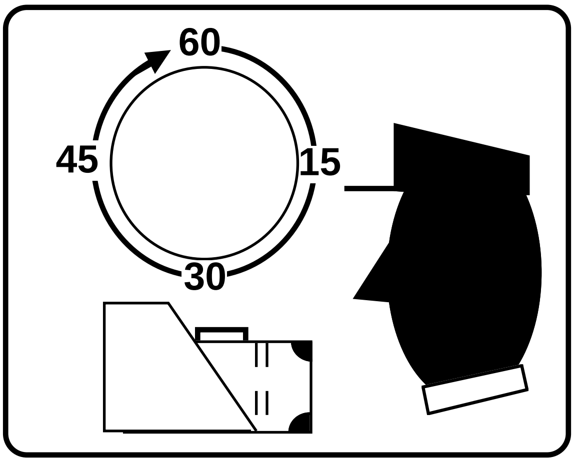
Neil hmm ... ok, try this - a sort of pie segment. The use of a shaded area is a sort of visual metaphor for a chunk of time – rather than just a point in time.





Mike Better... but I'm still not convinced it's right; it's still open to misinterpretation as something to do with 12 and 1 o'clock... the confusion with 1 0'clock comes partly from having two hands, which we've now eliminated, but also from the position of the number one, which is still in the position you'd expect to find it on a clock... but how can we avoid that. Putting the one at the twelve position like we tried just confuses the issue.
Got it! Use numbers 1 to 60 to show minutes passing.
Nobody'd think it means 60 seconds - that's ridiculous. Then we get away from this mixup between 1 hour and 1 o'clock

Neil that's a lot better - ok, if I do a final version ... I think we've made it.



APPENDIX 6: QOC TRAINING EXERCISE 3 – SCRIPTED VIDEO [STUDY 3]

The Design Session You About to Watch...

Background

Video recorders have a reputation for being overcomplicated to program, such as trying to set the day, channel, and times for recording in advance. Unfortunately, these usability problems are emphasised for elderly users, who represent a significant section of the public who use video recorders..

As part of my research project, a design session between two designers was recorded, as they discussed the design of a video player targetted at retirement homes for the elderly and physically handicapped. The discussion considers various design factors to help the elderly in programming the video player themselves, rather than rely on staff to do it for them.

Your job

— is to record the ideas and arguments the designers have, in the form of Issues, Options and Criteria.

The video is in two halves of about 7 minutes each. The designers talk quite quickly, so you will probably want to make brief notes of what they say, trying to spot Issues, Options and Criteria as they arise. Then you'll have 10 minutes to organise these notationally, as you've already been doing. You'll also do this for the second half of video. Up to this point, you'll be working separately.

Then you'll come together to compare notes in order to sort out any differences between your notations, and produce a final version which you think captures most accurately the arguments in the video. This stage will be 20 minutes long, and will be videoed. You will also be given copies of the sketches the designers make for reference.

The annotated transcript below shows the QOC elements embedded in the script of the video.

Text in **bold parentheses** are directions to the actors to add something to the sketches.

		Questions *(Question) = implicit Question*	**Options**	**Criteria**
Jamie	Right, we're clearly working under a few constraints here. One thing to focus on is the whole visual impact of it. Considering who the end-users are, if the controls look terrifying we've failed from the word go.			
Jim	definitely. Ok, let's look at some of the problems - we don't need to make any decisions yet, just get a feel for the design space we're dealing with here. Presumably we've got to have a remote control, as bending down to read the machine is out of the question.	*(what input device)*	*remote control*	*can't bend down*
Jamie	...unless we try some sort of speech recognition device, but the technology's too unreliable at the moment. So a remote control looks best – with its own display – something like this I guess **(Outline of calculator-style device)**. It'll have to have obvious keys like Play, Stop, Reverse, etc, and channel numbers, volume, brightness, teletext... plus programming commands like Start Time and Channel...	*(what keys on remote?)*	*speech recognition* *remote with display* *tape keys* *picture keys* *channel numbers* *programming keys* *teletext*	*poor technology*
Jim	hang on - how many old people need Ceefax control? Seems a bit unnecessary - we've got to think about really keeping functions to the simplest			*required functionality* *simplicity of function*
Jamie	won't subtitles and news be useful to them?			*required functionality*
Jim	well I suppose so, but can they read it?			*legibility*
Jamie	yeah, clarity's crucial here - I mean it's no good if they can't read the keys when they're programming	*(how to read keys)*		
Jim	why not use largish keys - you know, different colours, shapes, big labels? A bit like on kiddies' computer games **(A few example keys)**		*distinctive keys - colours/shapes/sizes*	
Jamie	uh-huh - but if buttons get bigger that means fewer altogether, or the handheld remote turns into a laptop keyboard...!		*compactness* *no. keys available*	
Jim	hmmm - but the problem is how to get all the necessary controls in...			
Jamie	have to keep working on that one. OK, what about actually programming the video to record something in advance - the player needs various bits of info - things like a starting time, duration...	*(how to specify times)*	*start and duration*	
Jim	no, better to ask for a start time and a *stop* time, rather than its duration – that involves mental arithmetic. Just ask for two times and they can copy them direct from the paper or TV times.		*start and stop*	*mental workload*
Jamie	Yeah, that makes sense. Ok, so how does the user know *what* info to put in *when*? Hang on, we need to decide first how you switch into program-mode from normal mode.	*how to switch modes*		
Jim	well, we'll have to put a Mode key on the remote somewhere - its position'll depend on the other keys we use. I guess in a corner'll do. **(Mode key added in corner of first sketch)**	*(mode key location)*	*top left corner*	

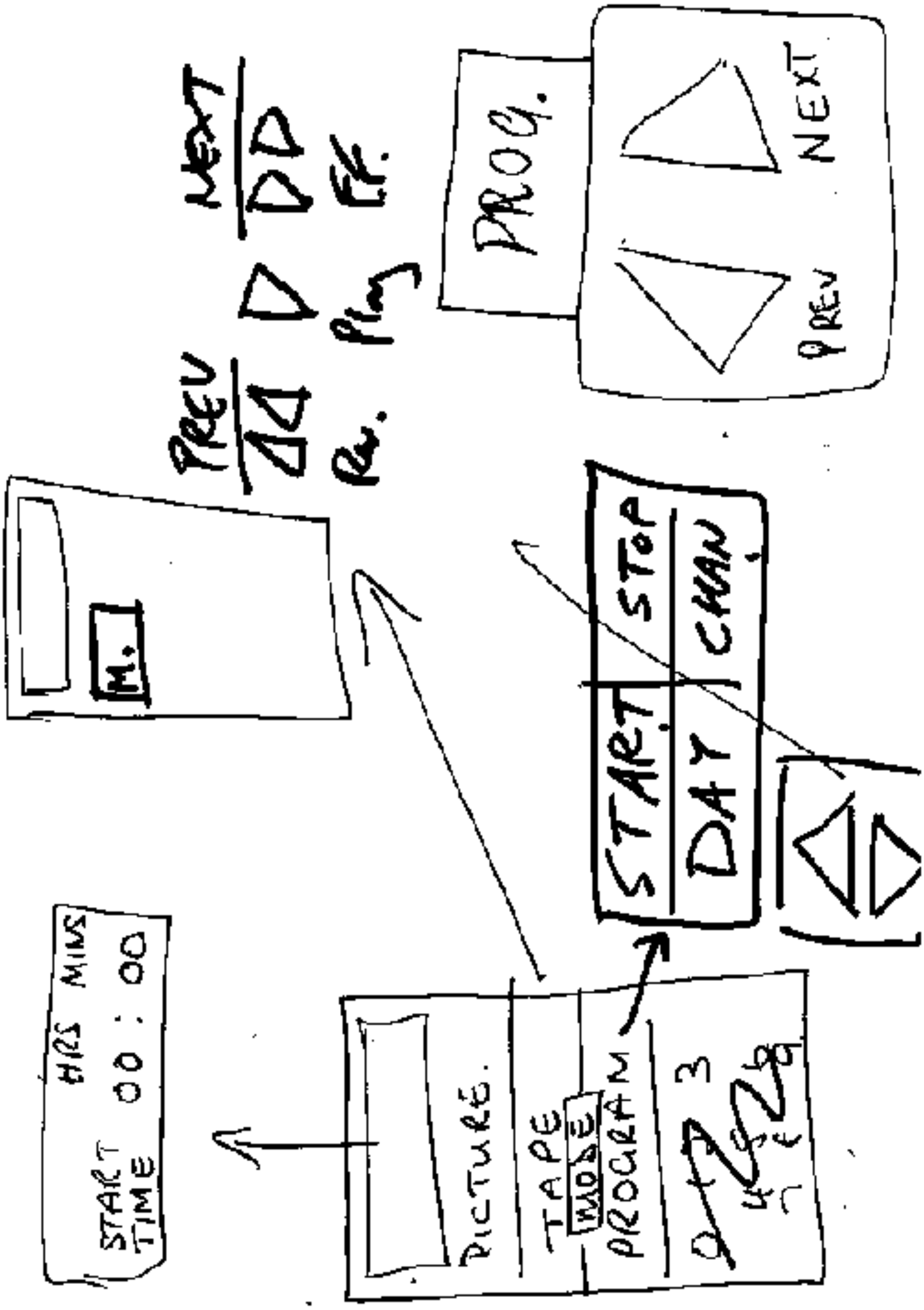
Jamie	Well we know we're going to have keys for programming and for tape control, so why not place it in-between to show that it sort of mediates between the two? Ok, what's all this going to look like? **(Sketch of remote: Picture controls under the display, then a section labelled 'Tape', and an arrow to the keys in No.2, then section labelled 'Programming')** ...and the Mode key goes here, in between Tape and Programming. Ok, so you'd hit the Mode key, and then press a key somewhere here for Start Time, and the display changes to prompt for a time... **(An enlarged display prompting for Start Time)** ...which you type in using the number keys - they'll have to go down here... **(Number keys 0-9 added to bottom of sketch of remote)** ...then do the same thing with Stop Time, then Day, and lastly Channel **(The four keys in the Programming section)**		*between key sections*	*function-location mapping*
Jim	Sounds straight forward enough, but channel should come first not last - more like the way we think normally, so mentally it should be less taxing	*(order of keys)*	*channel first*	*more natural*
Jamie	...If we've got keys to control the picture, plus numbers from 0-9 for channel and entering times, plus keys for the tape - fast forward etc, *and* the programming keys, that's a lot, especially if the keys are going to be large as we said, and the remote control not too bulky. The more keys, the more complex it looks...			*compactness*
Jim	ok – how about this: replace all the number keys with an up and a down arrow to increase and decrease numbers on the screen? **(Strikes out number-keys, and draws up/down arrow keys)**	*(how to enter numbers)*	*digit pad up/down arrows*	
Jamie	uh-huh, but then that takes a lot longer than just typing in the time directly.			*speed of entry*
Jim	the alternative is a much chunkier remote – I think it'll be hard to remove keys without sacrificing something - these old folk are after something simple to use, not maximum speed			*simplicity of function*
	B R E A K			
Jim	OK, where have we got to?			
Jamie	well, a big problem seems to be how to reduce clutter - of keys, and on the display. The size of the remote control and screen really makes things tricky.			*clutter*
Jim	hmm, ... at least one serious objection to using a small-screen remote is how do you review program settings?	*how to review programs*		
Jamie	yeah, it's rather awkward viewing Channel, then Start Time, then Stop Time, then Date in turn, for perhaps seven or eight different programs... you can only see one thing at a time, and they'll forget what they've seen and have to go back and check again. The only alternatives are to make the screen bigger, or the numbers smaller, neither of which are much good.		*bigger screen smaller characters*	*legibility compactness*
Jim	Well ... why don't we use the TV screen itself to display everything? With a screen that size, you could have 8 rows, and columns for day, channel etc, and see much more information. Something like this would really help memory. **(Table of columns)**	*(what display device?)*	*TV screen remote*	*amount of info displayed memory prompt*
Jamie	yes, we kill two birds with one stone: the numbers can be much bigger so you're not peering at this tiny little display on the remote, and the columns act as prompts to remind you to fill in all the bits of info, in case you forget.			*legibility prompting*

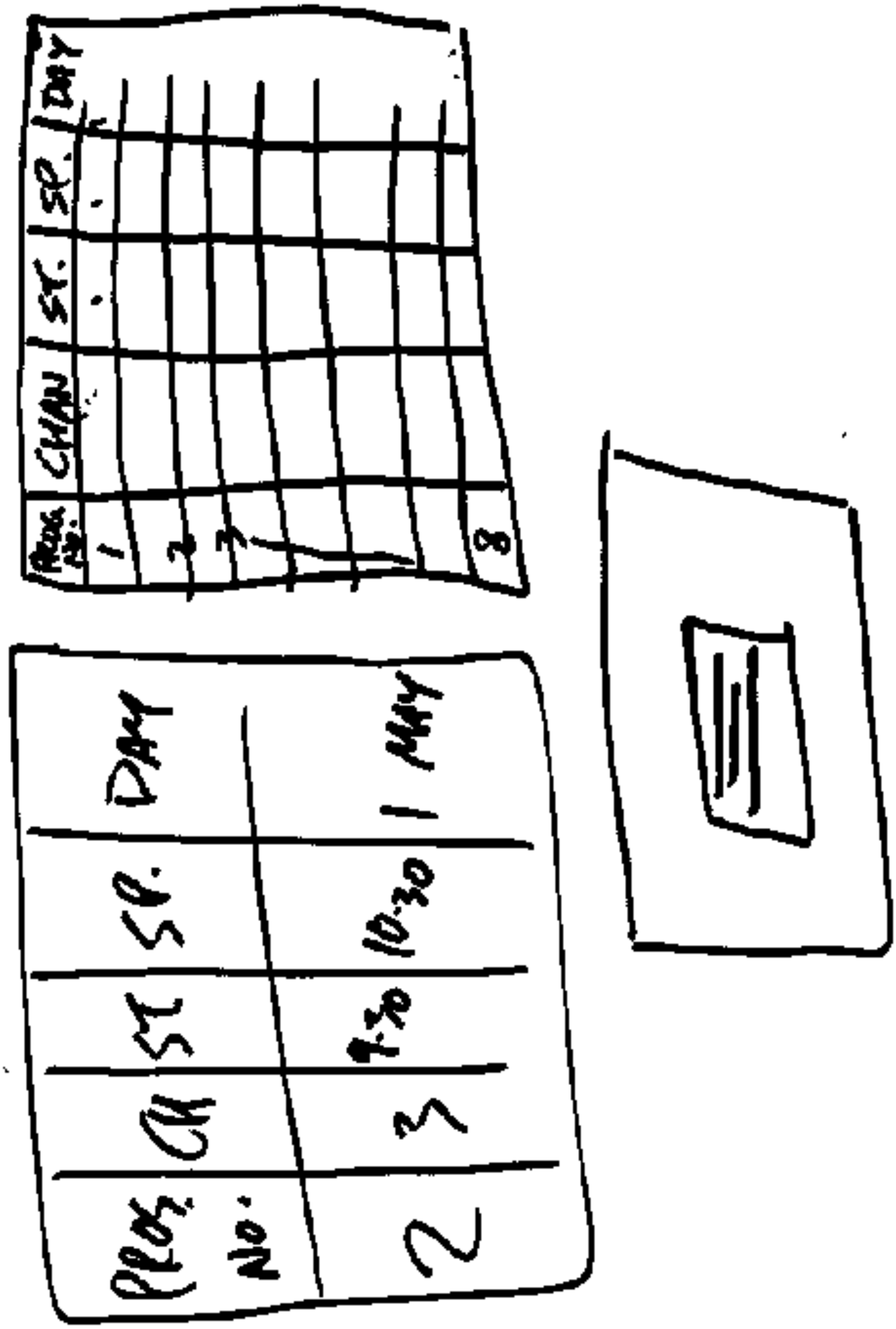
Jim	Ok, so how does it work at the moment? – to enter channel number, you push the Channel key, and then enter the time with the arrow keys. Next push Start Time, enter it, and so on...	*(how to switch columns)*	*special keys*	
Jamie	no this is no good—we've got to lose some of the keys - there're just too many Tell you what - we can get rid of the four programming keys and just have two keys for Next and Previous column, which jumps forwards and backwards between columns **(Left and right arrow keys for Previous and Next)**		*next/previou s keys*	
Jim	good one! ...well if we're really going to reduce the number of keys, is it possible to *reuse* some of the tape control keys - Fast-Forward and Rewind have arrows on them, and they could also control jumping to columns - saving us a couple more keys. **(Extra labels for Previous and Next above Rewind and F.Fwd keys in No.2)**		*reuse tape control keys*	*clutter*
Jamie	sure, but will the old folk understand the concept of modes - that one key can mean Fast-Forward *or* Next Column?			*intelligibility*
Jim	Mmm - maybe not.			
Jamie	Ok, so they have to understand *something* about modes to use the Mode key, but it doesn't actually change its *meaning* in the sense of switching from Fast-Forward to Next Column. I think we can go too far in reducing keys, and end up making the whole thing unintelligible.			
Jim	yeah ok - let's leave keys for Next and Previous in. Something we haven't looked at is what about when they make mistakes? Say they quit programming before they've filled in all the info? Seems to me that cutting errors should be a big goal for us - if they find they can't learn it at the first or second attempt, they'll just forget it	*premature quitting*		*reduce errors learnability*
Jamie	Perhaps the Mode button should only work if all the programs are either complete or empty - so half completed ones aren't allowed?		*mode key blocks incomplete entries*	
Jim	no they'd hate that - they couldn't change their minds half-way through. Why not prompt them? - I like this idea of prompts - they can be made nice and friendly: "You haven't entered a channel number - do you still want to quit?" or something. **(Prompt box appearing in the middle of the screen)**		*screen prompts*	*'friendliness'*
Jamie	sounds good - a green key on the remote for Yes and a red one for No? ...But that adds more keys...		*green yes key, red no key*	*key clutter*
Jim	well I suppose we could use the Previous and Next keys to mean No and Yes when messages like that popped up... but we're back to the modes problem again - they'll have to learn *what* those keys mean *when* and we want to avoid all possible learning problems - keep it nice and simple - I think we should forget prompts for the moment. Perhaps simply displaying these four columns will be enough to jog their memories		*reuse Prev/Next for Y/N* *display columns*	*mode confusion learnability* *memory load*
Jamie	yeah, too much hassle. I just wonder - is a screen filled with rows and columns of numbers going to be too confusing? And if you're down here at the bottom of the table, and you want to change the channel up here, how can you get up there without pressing the Previous key twenty times?	*(how to display program-ming info)*	*rows and columns*	*information clutter* *screen navigation*

Jim	Hmm...now that we're using the TV screen, seems like we're not maximising its main advantage - *big* characters. Couldn't we just display the four columns for one program at a time, in extra large characters? **(New display showing one program at a time)** ... and to edit another program you press a Program key to take you to the next one.	*how to switch progs.*	*one row display at a time* *Program key*	*legibility* *information clutter*
Jamie	uh-oh – another key			*key clutter*
Jim	well look at it this way: if we stick with the table display, we've got to speed up movement around the table, and the only way I can think of doing that, is with a key to jump between rows! You'd press it to jump to the next row, or hold it down to skip through rows quickly.			
Jamie	ok you win. **(Program key added above Previous and Next keys in No.10)**. I think we've got near optimal balance between legibility, how much info you display at once, and speed.			*legibility* *information content* *speed of entry*

APPENDIX 7: DESIGN SKETCHES IN THE SCRIPTED VIDEO [STUDY 3]

Below are the sketches which the designers produced in the scripted video (referred to in the script in Appendix 6). They are reproduced to convey the realism of the scripted video as a medium for communicating realistic design activity, i.e. dialogue coupled with sketching.




PROG. No.
CHAN
ST.
SP.
DAY
1
2
3
8
PROG. No.
CH
ST
SP.
DAY
2
3
9.30
10.30
1 MAY

APPENDIX 8: ATM DESIGN PROBLEM STATEMENTS [STUDY 3]

The 'Upstream' ATM problem

Problem statement is brief, **no cues provided** as to the relevant issues or criteria to consider, simulating the situation of designers just beginning to generate ideas over a new problem.

Design Task

The National Bank Automated Teller Machine (ATM) is a fairly typical ATM. If you want to get cash from it, you could go through the following steps:

- Push card into slot
- Type in PIN (Personal Identity Number) when prompted

  The screen then shows six services
- Select *Cash Withdrawal*

  The screen then shows five preset amounts and the option *Another Amount*
- Select *Another Amount*
- Type in the amount required and press the *Enter* key

  The screen then says "Do you want another service?"
- Select *No*
- Remove card from slot
- Take cash from drawer, and receipt from slot.

But...

The bank noticed that at certain times of the day, long queues built up at these ATMs.

Your task...

You are brought in as design consultants by the National Bank, who ask you to design a new one to serve the customers better.

Summarise your new machine at the end of the hour.

The reasons behind the design should all be recorded in the form of Issues, Options and Criteria, and supplied along with the final design.

You are free to use drawings or notes throughout your work.

The 'Downstream' ATM problem

In addition to description of current design, proposes **an alternative design** for consideration, plus **additional information on problems with the existing design —** cue subjects to pertinent issues, design alternatives and trade-offs.

Design Task

The National Bank Automated Teller Machine (ATM) is a fairly typical ATM. If you want to get cash from it, you could go through the following steps:

- Push card into slot
- Type in PIN (Personal Identity Number) when prompted

    The screen then shows six services
- Select *Cash Withdrawal*

    The screen then shows five preset amounts and the option *Another Amount*
- Select *Another Amount*
- Type in the amount required and press the *Enter* key

    The screen then says *"Do you want another service?"*
- Select *No*
- Remove card from slot
- Take cash from drawer, and receipt from slot.

But...

The bank noticed that at certain times of the day, long queues built up at these ATMs.

The Fast ATM (FATM)

The National Bank asked their own design staff to see if they could speed the process up. Their proposed design (called FATM) presents the customer with the following procedure:

- Select cash amount (one of three preset amounts)
- Insert card
- Remove card
- Type in PIN
- Take cash and receipt from drawer

Shortly after this, the bank's Customer Interest Group reported independent research into the effectiveness of the ATMs.

They found that:

- multiple requests for services often caused queues
- customers often forgot to take their receipt
- over half of first time users of the ATM made errors
- most customers said they liked having a variety of services at ATMs

Your task...

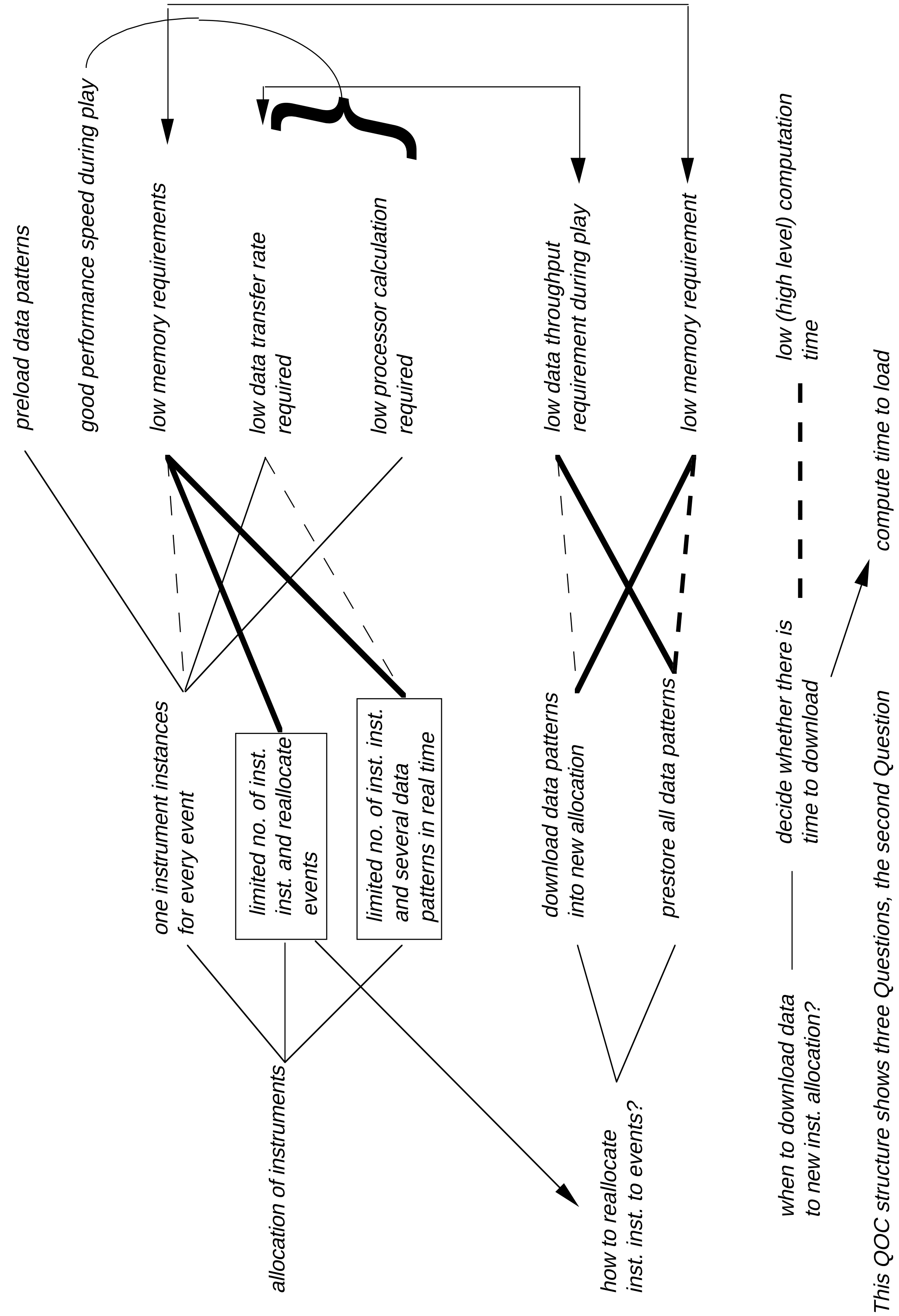
You are brought in as design consultants by the National Bank, who would like to know whether you think the FATM is a successful design, and if necessary to design a new one to serve the customers better.

Summarise your machine at the end of the hour.

All the reasoning behind your final design should be recorded in the form of Issues, Options and Criteria, and supplied along with the final design.

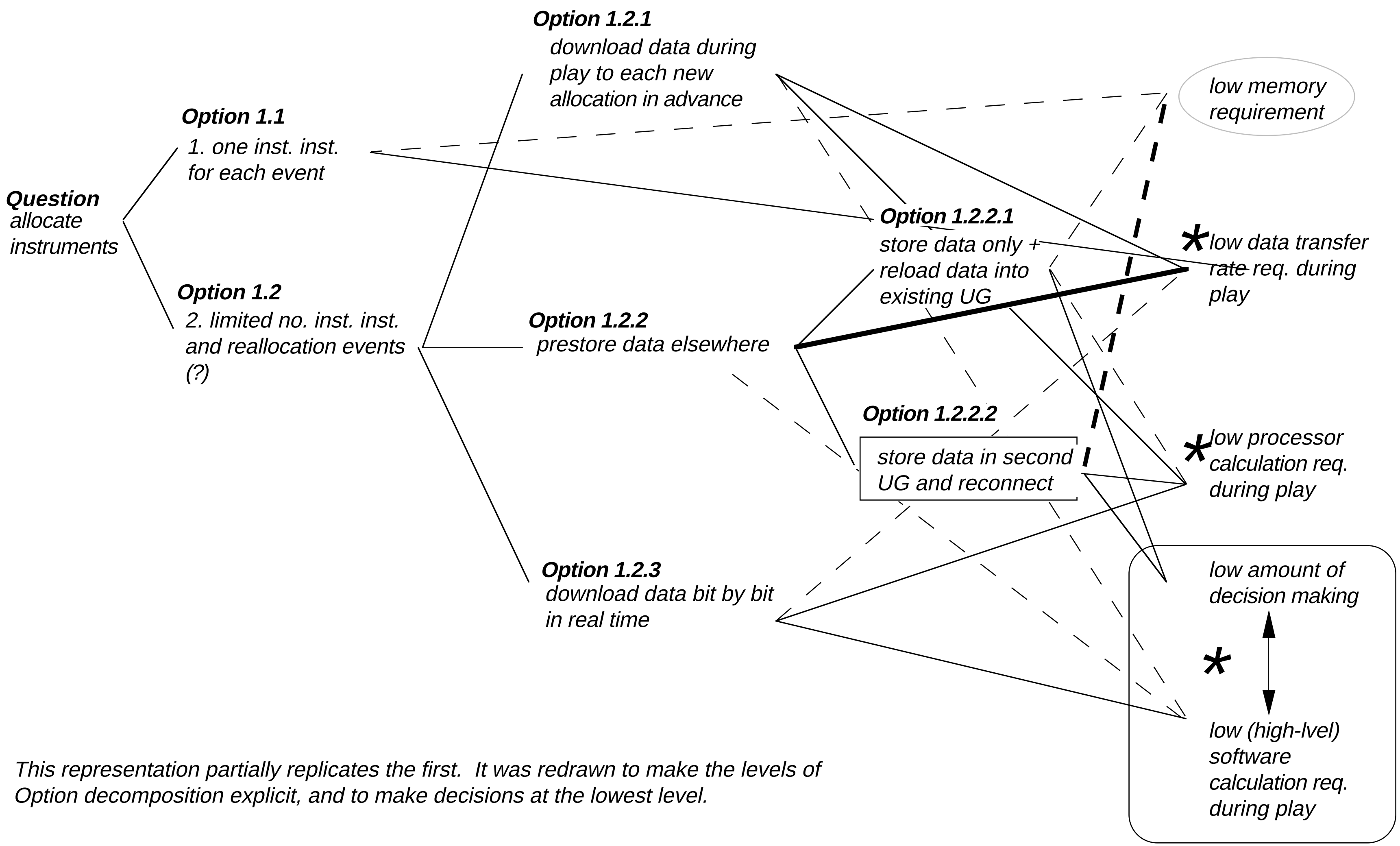
You are free to use drawings or notes throughout your work.

APPENDIX 9: QOC REPRESENTATIONS FROM CASE STUDY 1



This QOC structure shows three Questions, the second Question consequent to the first. The designer also marked Criteria in the consequent Question which were inherited from the first.

(Option labels have been added to show their decomposition)



This representation partially replicates the first. It was redrawn to make the levels of Option decomposition explicit, and to make decisions at the lowest level.

APPENDIX 10: OUTLINE OF REVISED QOC TUTORIAL FROM CASE STUDY 2

The top-level outline of the tutorial is shown on the left, with elaboration where necessary.

Topic heading	***Details***
Why Design Rationale?	introductory orientation
Design Rationale: Our Research Strategy	the DSA approach to DR
Characterising Design Practice	*Communication of information between teams* *Collaboration among individuals in team* *Problems :* *Impoverished Communication* *Impoverished collaborative problem solving*
Representing the *Design Space*	basic constructs of QOC
Why QOC?	aiming for balance of useful structure and usability
Addressing the problems with QOC	how QOC can help impoverished communication and impoverished collaborative problem solving
QOC Notational Conventions	*Questions : The key issues in the design* *Options : Possible design decisions* *Criteria : The goals and constraints on which decisions are based* *Assessments : The evaluation of Options against Criteria* *Assumptions : Delimit the scope of the QOC*
A QOC Design Process	initial mention of the 5-phase process model
Heuristics for Using QOC Representation to Augment the Design Space	initial mention of the 8 heuristics for DSA (derived from properties of the notation)
A Scroll Bar Example (Detailed design)	small-scale example of the QOC process model phases in use
An ATM Re-Design Problem	second example, illustrating phases of process model, and heuristics for structuring the QOC
Focussing Questions	examples within the ATM problem of poorly focussed Questions - too narrow or too diffuse.
“Diffuse” Question Solution	how to remedy an unfocussed Question - decompose into separate Questions
A “Non-DR”	example of a poor Question (about a Criterion) and explanation of weaknesses

QOC Notational Conventions	recap of QOC's main constructs
How to Design with QOC	detailed description of what each phase in the process model involves: *Phase 1: Get relevant information down* *Phase 2: Structure material into rough QOC* *Phase 3: Flesh out design space* *Phase 4: Reformulate design space to tidy it up.* *Phase 5: Make design decisions*
Heuristics	as process model is not meant to be strictly linear, heuristics can be applied at any point
Some Common Problems & Some Hints	*Don't get hung up on the QOC notation* *Don't worry about making decisions too soon.* *A rough QOC may need to be radically transformed.* *Questions may need to be refocussed* *Don't record minor refinements of ideas as different Options (or Q, C)* *Issues identified early are not necessarily good Questions.* *You won't follow the phases exactly as suggested*
Tutorial ATM Re-Design 1	small ATM redesign scenario, and suggested QOC
Tutorial ATM Re-Design 2	second ATM redesign scenario and QOC

APPENDIX 11: QOC CRIB-SHEET USED BY DESIGNERS IN CASE STUDY 2

QOC Design Rationale

Crib-Sheet

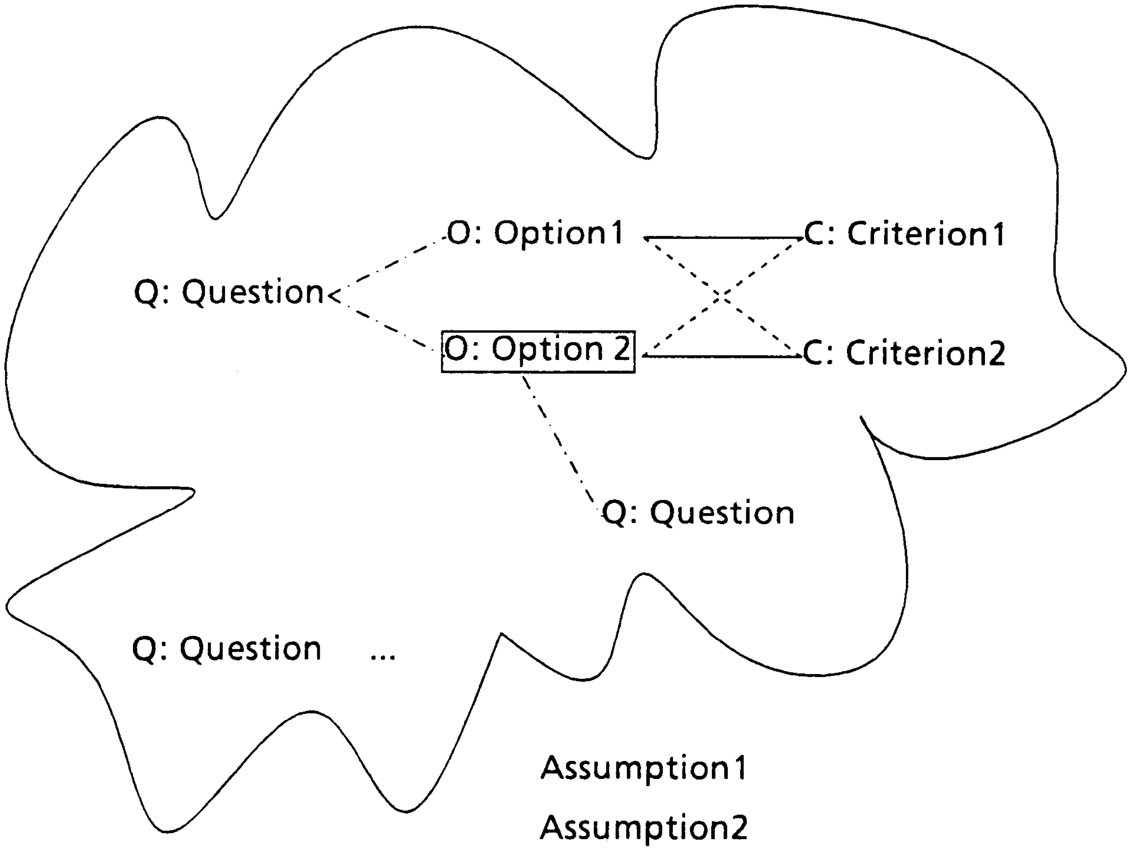
Allan MacLean, Simon Shum,[1] Victoria Bellotti

Rank Xerox EuroPARC

[1] also University of York

- ❏ Notational conventions
- ❏ How to design with QOC
- ❏ Some common problems

QOC Design Rationale Notational Conventions



Questions: The key issues in the design
An Option may suggest a *Consequent Question*
Several independent Questions may be formulated

Options: Possible design decisions
A boxed Option is a chosen *decision* (or current preferred choice)
An unboxed one is a possible *alternative*

Criteria The goals and constraints on which decisions are based
Criteria are always worded as "good" - e.g. *low errors* rather than *errors* ; *simplicity* rather than *complexity* .

Assessments: The evaluation of Options against Criteria
A solid line is a positive Assessment - the Criterion supports the Option
A dotted line is a negative Assessment - the Criterion objects to the Option
(Strength of Assessment may be indicated - e.g by a double line)

Assumptions: Other decisions or constraints which delimit the scope of the QOC

How to Design with QOC

Phases

Phase 1	**Get relevant information down**
Tasks	Get a feel for the main issues Work out what information provided is relevant (& classify as Q,O C if possible)
Phase 2	**Structure material into rough QOC**
Tasks	Structure and make sense of the information available Find good Questions
Phase 3	**Flesh out design space**
Tasks	Use current understanding of design to help generate new ideas. Generate new Options Generate new Criteria
Phase 4	**Reformulate design space to tidy it up**
Tasks	Tidy up description and make it more coherent Reword Q, O, C if necessary Reformulate Questions (and reorganise O, C) to improve decomposition
Phase 5	**Make design decisions**
Tasks	Evaluate and select Options (i.e. draw boxes around decisions) (Use Criteria to evaluate Options) (The level of detail represented may not include all relevant information ± you may need to consider the importance of Criteria, or their assessment, or interdependencies with other parts of the space.)

Heuristics

1. Use Options to generate Questions

2. Use Questions to generate Options

3. Use Criteria to generate Options

4. Use Options to generate Criteria

5. Consider Extreme Options

6. Consider Distinctive Options

7. Represent Positive and Negative Criteria

8. Overcome negative Assessments, maintain positive

Some Common Problems & Some Hints

Don't get hung up on the QOC notation

If you are stuck move on and come back to the problem later

It can make sense to spend time discussing the problem informally to help understand the real issues, and then use QOC to reflect on your discussions.

Don't worry about making decisions too soon.

QOC helps by keeping things on the table so that you can decide later when you understand the problem better.

A rough QOC may need to be radically transformed.

Questions may really be Criteria

Options may be Criteria

Options may refer to other Questions

Questions may need to be refocussed

Pull multiple issues into separate Questions

Ask a more general or more specific Questions to get an appropriate focus (e.g. - don't ask 'yes/no' Questions)

Don't record minor refinements of ideas as different Options (or Q or C)

Preserve only the version you think is most appropriate.

Issues identified early are not necessarily good Questions.

It can pay to ask if you are asking the right Question.

You won't follow the phases exactly as suggested

You will have some good ideas you want to preserve while you are still organising the initial information

You will have some thoughts or see something in the design requirements which is relevant when you think you are tidying up things

The Heuristics may be applied opportunistically at any phase of design.

APPENDIX 12: DESIGN PROBLEMS USED IN CASE STUDY 2

People and Objects Environment:
Computer Supported Cooperative Work (CSCW)

POE is a CSCW environment for integrated multimedia workstations which is being designed by your company. It supports production of multimedia brochures for use in large international holiday companies. POE stores multimedia articles, used in brochures, in files called ‘Objects’ and represents users as ‘People.’ It also supports video-phonecalls between People. There are two other important features of POE:

- An Object can only be possessed by one Person at a time.
- A Person can gain possession of as many free Objects as they want and display them in separate editing windows (not shown in figures 1 or 2). The only way for others to edit one of these is to ask the Person to give up that Object.

The information about the current status of People and Objects in the environment is presented in a shared desktop-view window which is the same for all POE users.

People are represented in a window as one of three different coloured head and shoulders icons. Each icon has a Person’s name label.

- *Black* is ‘yourself’, whatever your current Object ownership status is.
- *White* represents current possessors of Objects.
- *Grey* represents POE users who do not possess an Object but can still be video-phoned.

Users can select a Person’s icon and click on a ‘make call’ button to make a video-phonecall .

Objects in POE are represented as icons spread around the desktop, and uniquely identified by a name-label (their system filename). Only the current version of an Object is displayed and you need to select it to edit it. The appearance of an Object icon identifies the type of medium it contains (e.g. audio, still photograph, or text). The icons of Objects which are busy (inaccessible because they are currently possessed by another Person)are displayed with a rounded border.

Research into target users’ existing practices shows that they use the videophone facility to talk to one another a great deal and pass round and collaboratively edit articles and media which make up brochures, so they need information about:

- What items my colleagues are working on which I might want to discuss with them.
- Who has the current version of a particular item which I want to edit now.

Two possible designs for displaying People and Object status information are being considered for their different merits and the kinds of task they support best:

a) *People and Objects Overview* (Figure 1): Person icons are only represented next to the icon of the Object that they are currently editing (inside the rounded 'busy' border). So some Objects are displayed as busy but with no feedback as to who is using them. People switch from editing one Object to another by clicking in the appropriate editing window, so Person icons tend to move unpredictably around the screen from one boxed Object to another.

b) *People in Own Separate Overview* (Figure 2): Person icons are represented in a separate overview and Objects are displayed without feedback as to who is using them. To find out what Objects a person is using, the appropriate Person icon is selected. This highlights those Objects possessed by the selected Person.

Your task

You are brought in as consultants to evaluate, and possibly improve on the two proposed POE Display designs, a and b. The target users will want to communicate and coordinate their activities with those of other people and keep track of the progress of work objects they are interested in. The designs need to be evaluated with particular attention to whether they will support this kind of work.

Use QOC to help organise your design deliberations. You should aim to produce a Design Rationale to summarise your conclusions to your company. It should include:

- Representations of the major points you discuss.
- Reasons for your conclusions about the proposed designs.
- Justifications for your recommended solution.

The following points should help to guide you in this exercise.

- Identify the major issues at stake in the POE Status Display design.
- As a hint, supporting the necessary tasks is likely to be one source of Criteria
- Evaluate the two proposed designs using QOC notation.
- Suggest any alternatives which you think may be worth considering.
- Recommend a design and highlight your choices within your rationale.

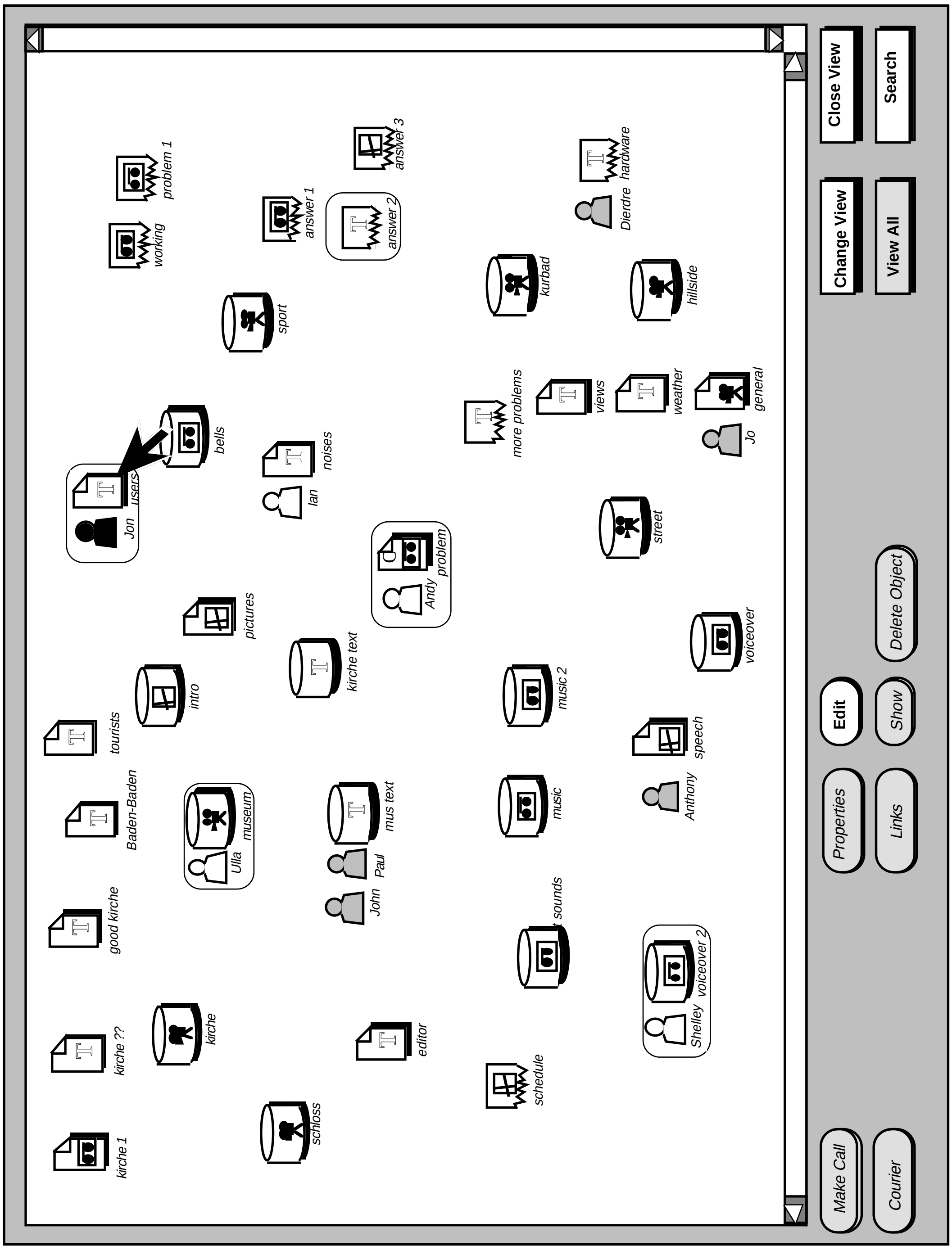


Figure 1: Suggested design for option (a) with person icons included in 'object overview'

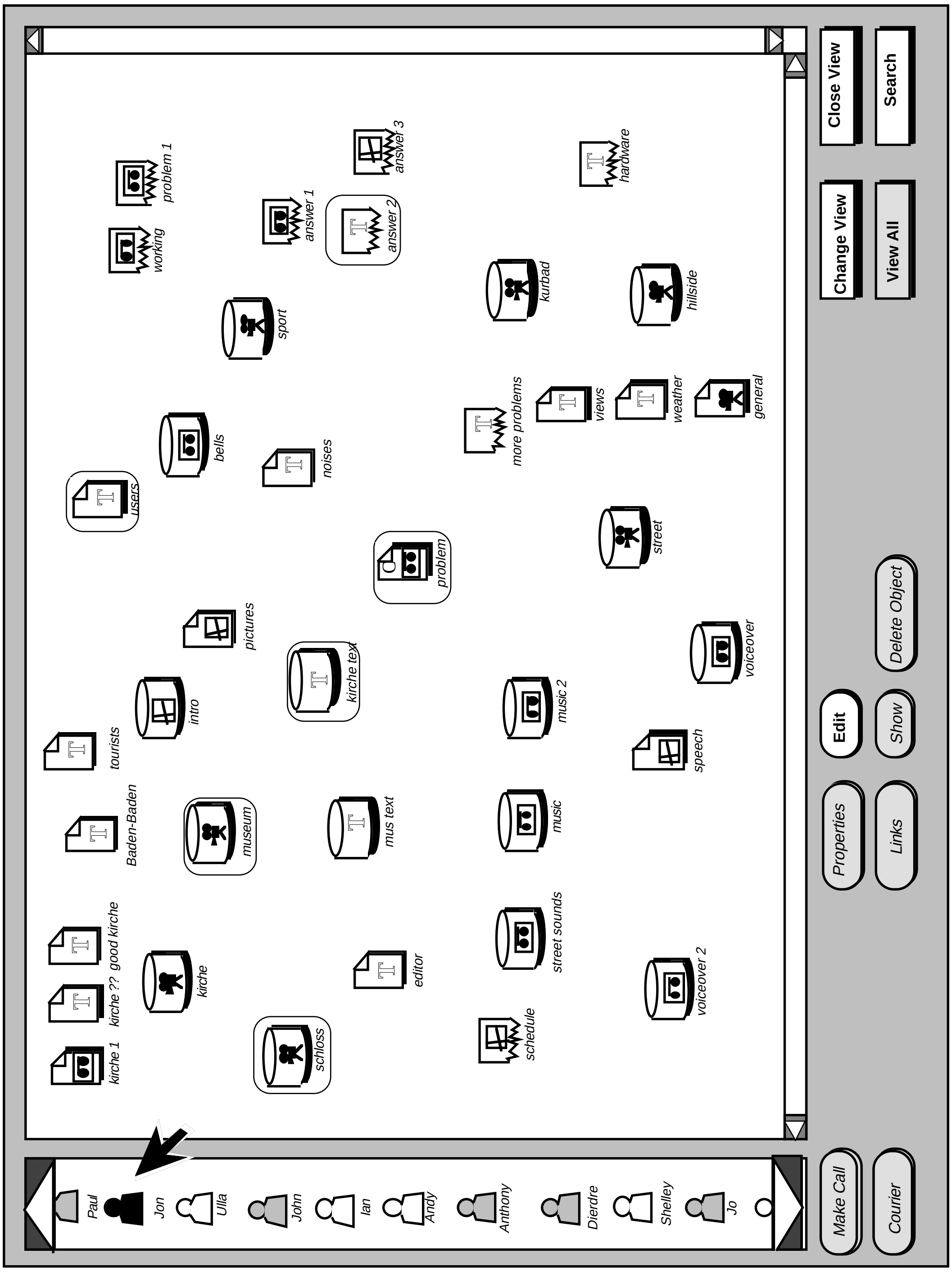


Figure 2: Suggested design for option (b) with the person icons in a separate 'workgroup overview'

NetGroup design problem

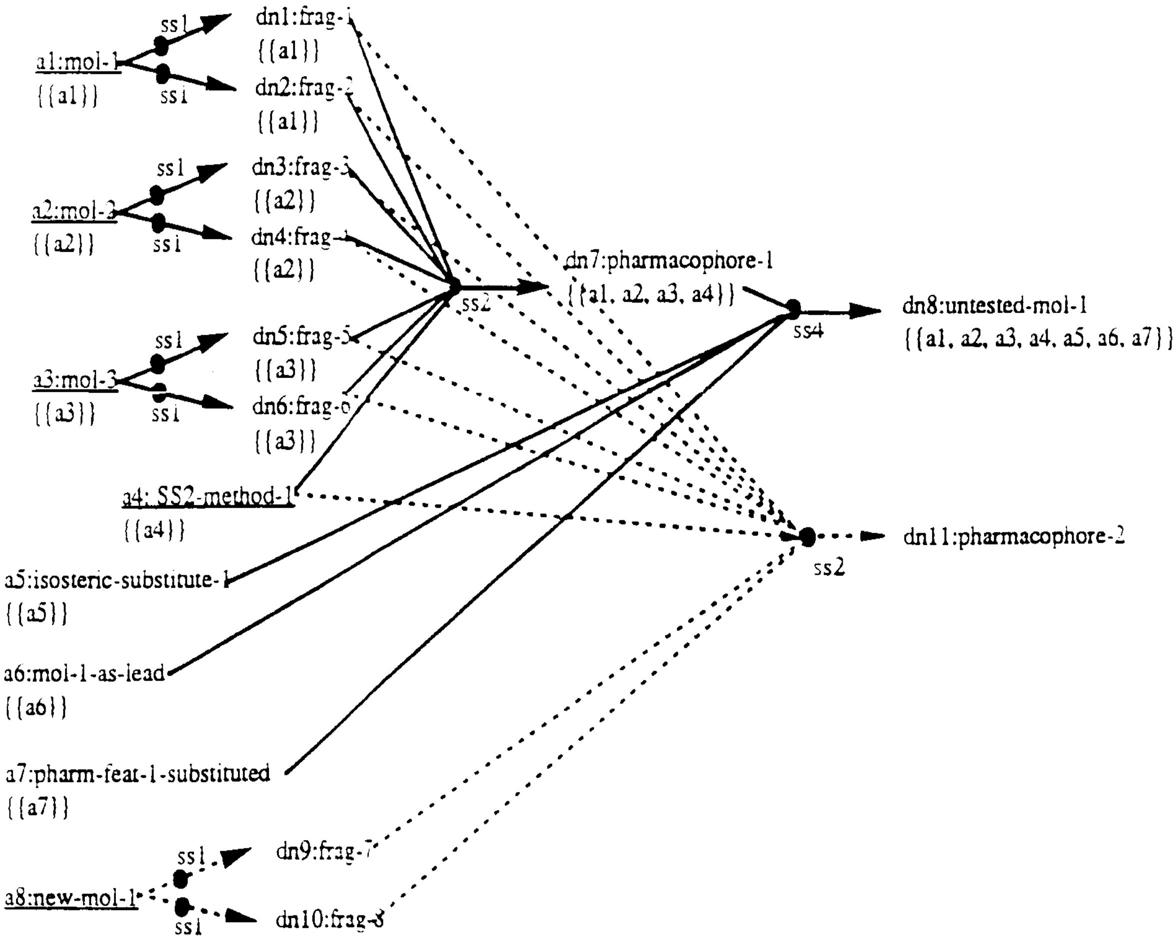
The NetGroup project is concerned with supporting drug design. The current prototype system may be divided into three main support systems.

SS1: Responsible for analysing a chemical compound and breaking it into its smallest useful building blocks. These are known as *molecular fragments.*

SS2: Compares the fragments produced by SS1 from different molecules in an attempt to produce a model which describes common features of all the molecules. This model is called a *pharmacophore* and may be generated using different methods.

SS3: Structure activity relations. Allows the user to compare different chemical structures to determine their effect on the overall activity of a molecule. *Note:* the effects of this support system are not relevant to this problem.

Whilst reasoning about drug design, or indeed about any type of design, it is essential to keep track of the dependencies between the objects which are generated. For example, fragments 1 and 2 may be dependent on molecule 1 (from which they were created) and pharmacophore 1 may depend on fragments 1 to 6 as well as a method for its generation. In the second prototype we intend to represent this information using a truth maintenance system (TMS) where all the dependency information is represented in a network like the one below.



In such a network, the nodes represent objects or pieces of information and the arcs show the dependencies. As can be seen, the network is annotated with essential textual information.

These networks can grow to be very large and the problem we would like to address is how to allow the user to obtain information from the whole network when it will not all fit on the screen at once. We are prepared to consider using an alternative representation for the net, so long as it still provides the same information.

Context

System: Mac IIci with screen size 7"x9" screen. Input devices are mouse and keyboard. Colour is available, and graphics may be made active (clickable)

Target users: System developers. The end user would not be subjected to the full detail of the TMS network.

Tasks: Determining dependencies by tracing routes through the network, studying the textual annotations on the net.

Design situation: Time restrictions on the software development would not allow implementation of more than one prototype.

FileGroup design problem

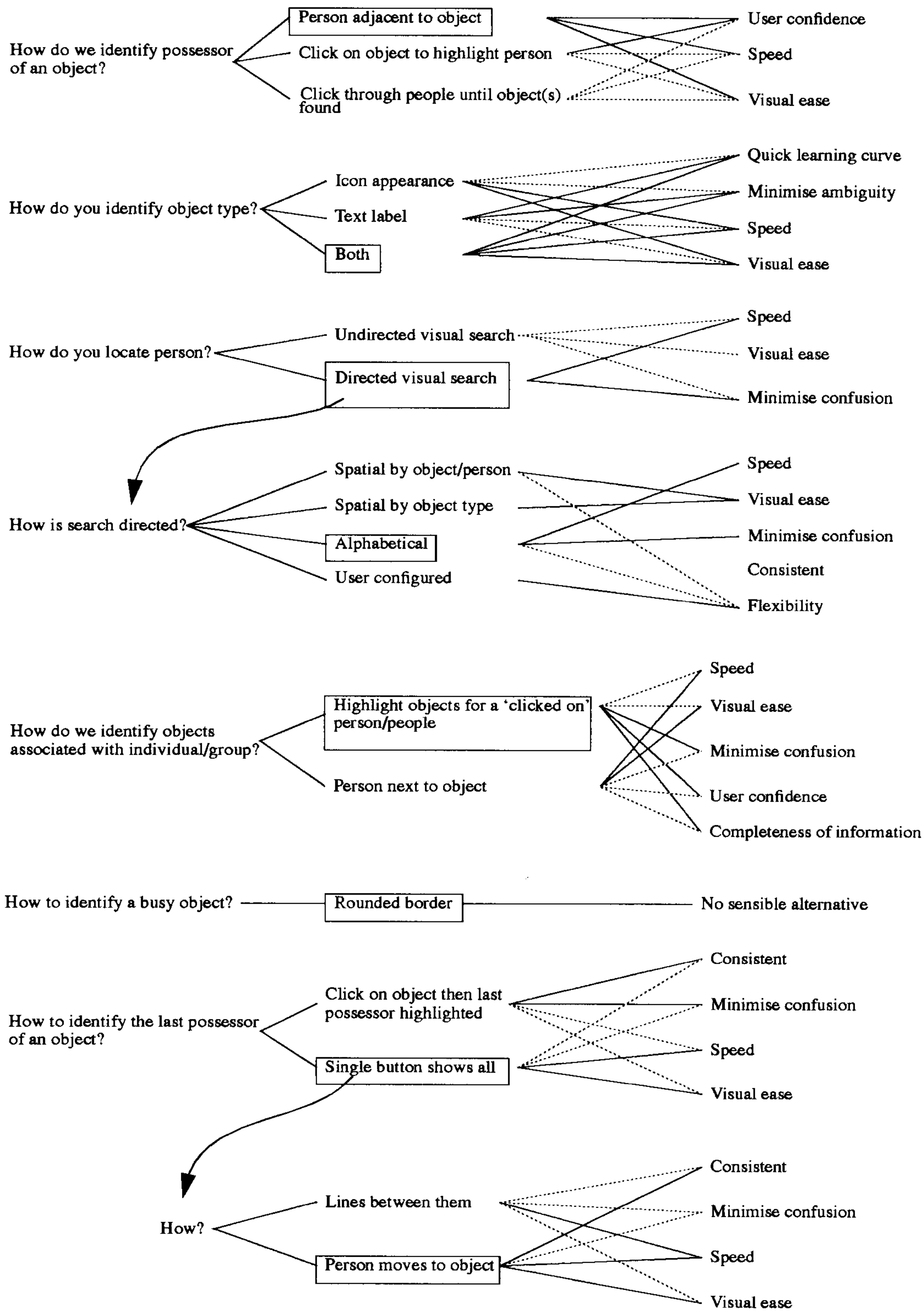
This problem cannot be detailed for reasons of confidentiality. It is described in Case Study 2 as follows:

FileGroup were concerned with designing the optimal file format for data files. Their declared requirements were to minimise disk space, be able to check the consistency of files, and to store sufficient information within a data file (the specific details of this work is confidential, but the essence of the problem is sufficient for present purposes). The designers had specified two clear issues to resolve in the exercise, which were to decide on a file format, and to design the file header which carried the information needed to interpret the data in the file.

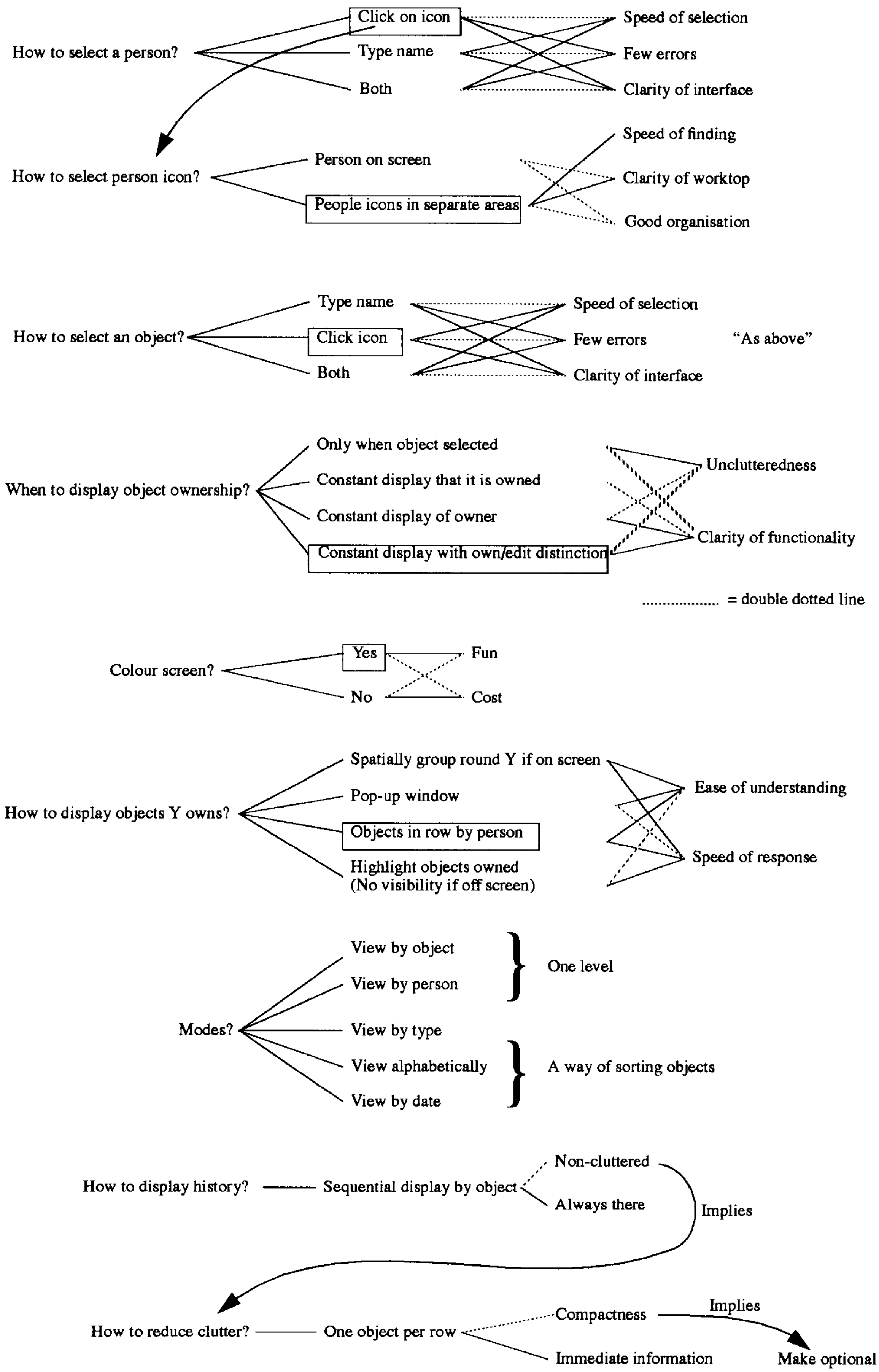
APPENDIX 13: QOC REPRESENTATIONS FROM CASE STUDY 2

Reproduced are the QOC representations generated by each of the two design teams (NetGroup and FileGroup), for each of the two problems (P&O problem, and their own selected problem). These QOCs are taken from Elworthy's (1991) analysis of the sessions.

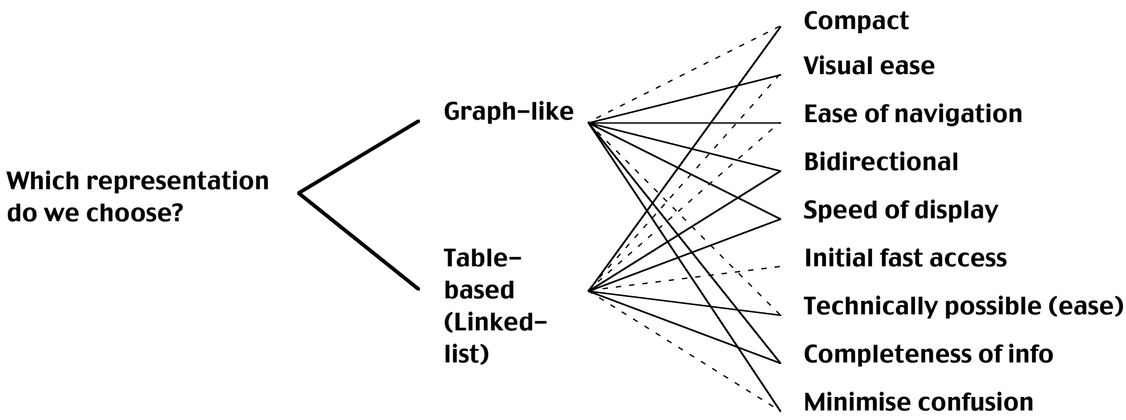
NetGroup: QOC for P&O problem

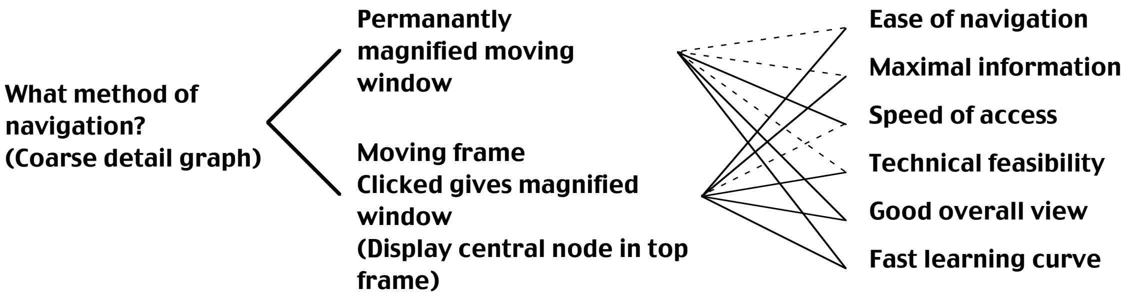


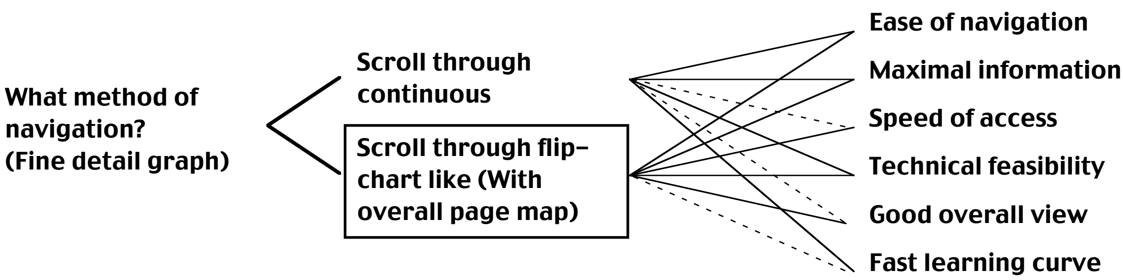
FileGroup: QOC for P&O problem



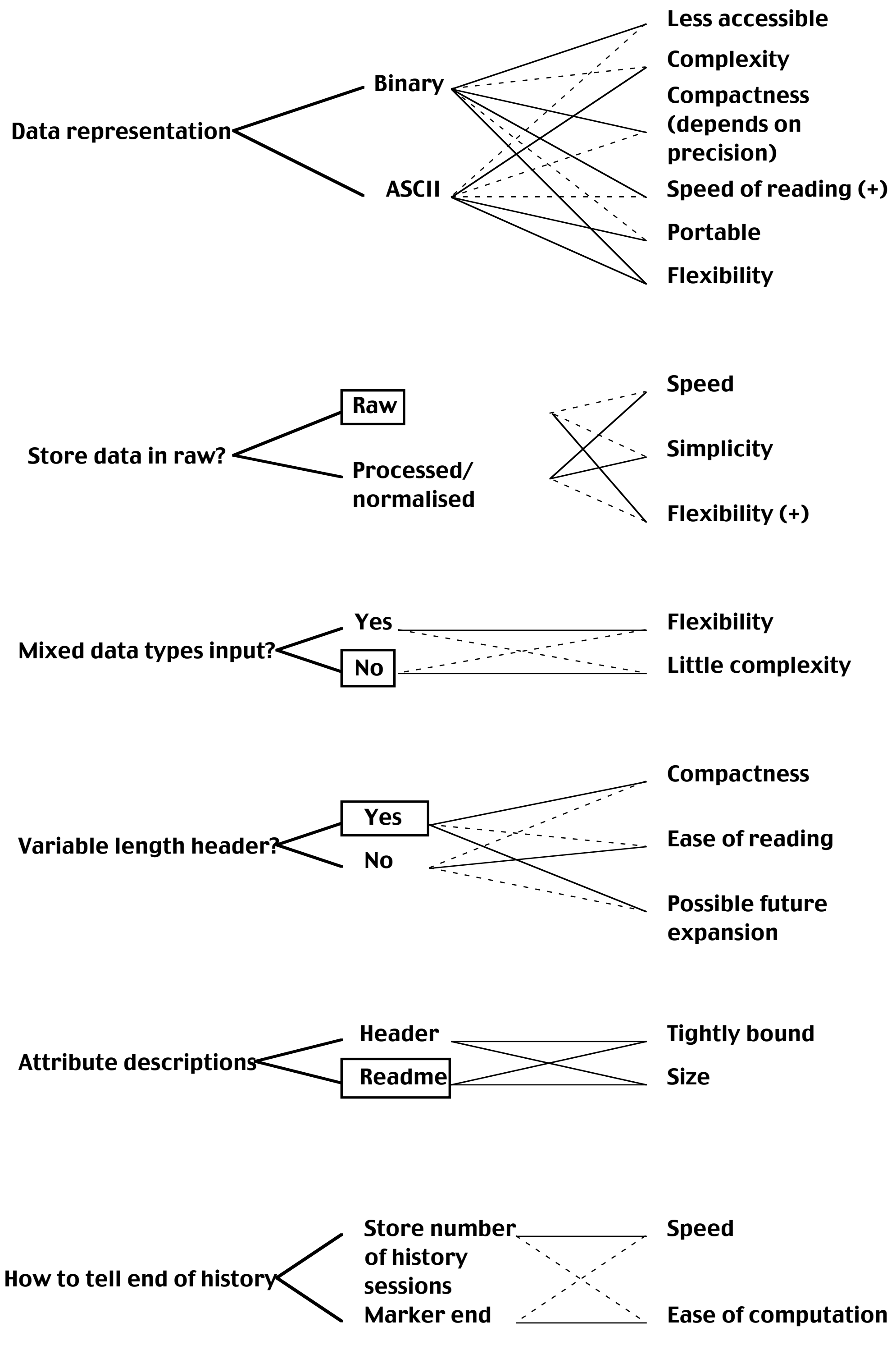
NetGroup: QOC for own problem







FileGroup: QOC for own problem



APPENDIX 14: 'QOC HANDBOOK' USED IN CASE STUDY 3

QOC Design Rationale: User's Handbook

(version 1.0)

The purpose of this handbook

This is meant to be a practical guide to using the design rationale notation *QOC*, being developed by Rank Xerox at the Cambridge EuroPARC research lab. QOC stands for *Questions, Options* and *Criteria,* the main elements in the notation. The conceptual basis for creating design rationale (DR) along with the actual design can be found elsewhere (see bibliography at end), and so only a brief introduction is provided here.

QOC has been under development for about 3 years now. To date, its users have been members of the research community, either developing it or using it experimentally on small scale design projects. As such, many of the principles in this handbook have been gleaned from users relatively expert in the notation, from their personal experiences, and from lab-based studies of QOC's use by programmers and other members of the IT community. It is hoped that as QOC is used more extensively, users' experiences will be incorporated into this handbook, and updates released periodically.

Background: The need for design rationale

It is being realised that the design process could be helped considerably if at the end there was not only a **design** (e.g. a piece of software), but some way of understanding *why* it is the way it is. System maintainers spend a lot of time trying to understand a system before they can do anything to it. A **design rationale** is one way to "get inside the designers' minds" in order to see where they were coming from when they made their decisions. It is a move away from the *product* oriented design practices used at present, which focus on the creation and tracking of various design documents, and adopting instead a more *process* oriented approach, in which information which answers *why* and *how* questions about design decisions is also considered of prime importance.

To represent DR, we need a simple notation, amenable to tool support, which can be learned relatively quickly, and isn't too intrusive to ongoing discussion. In addition, in order to offer the designers some payback for their trouble, using the notation during design meetings/on their own should actually help the problem solving process in some way, ie. you benefit from recording your ideas in the notation.

As EuroPARC's work is in the area of Human-Computer Interaction, much of the QOC work has focussed on representing user interface design reasoning, and one way in which QOC is envisaged is as a means of encouraging designers to consider human factors during design work. However, QOC (and DR in general) is by no means limited to these issues, as examples below demonstrate.

Introducing QOC: a design rationale notation

There are a number of DR notations under development by various groups in the UK and US. All of them have node and link types to allow you to record the subissue being dealt with, the alternative courses of action open, and the criteria for choosing between them. QOC is one of the simplest of the 3-4 available notations. Design reasoning is represented using the following entities and relations (with abbreviations):

Question (Q)

– a brief summary of the problem, phrased as a question in order to encourage consideration of what the real issue is

Option (O)

– an alternative solution which could be pursued

Criterion (C)

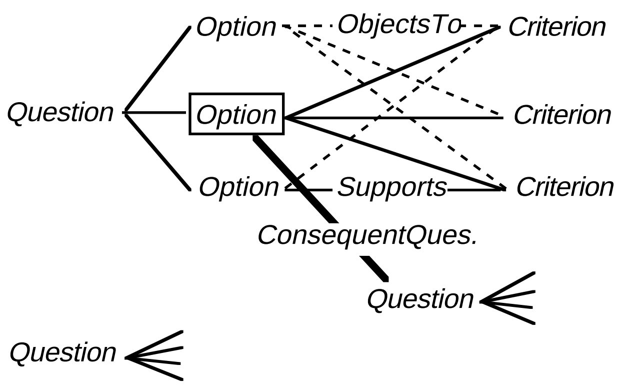
– a reason/design principle which supports or objects to each Option

Supports and ObjectsTo relations

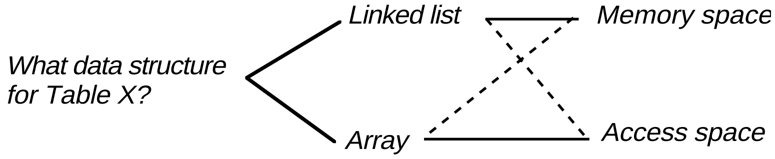
– links connecting Criteria to Options

QOC graphical structure, and examples

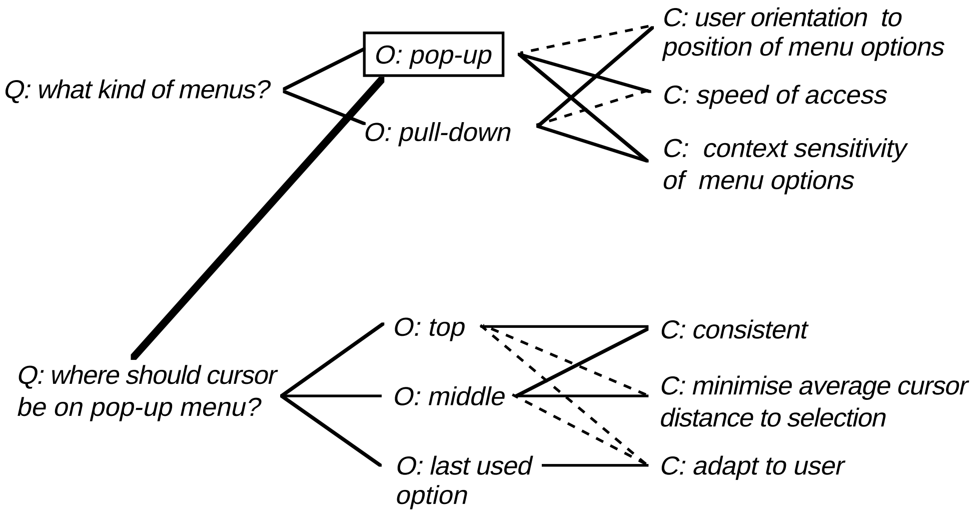
Generic structure



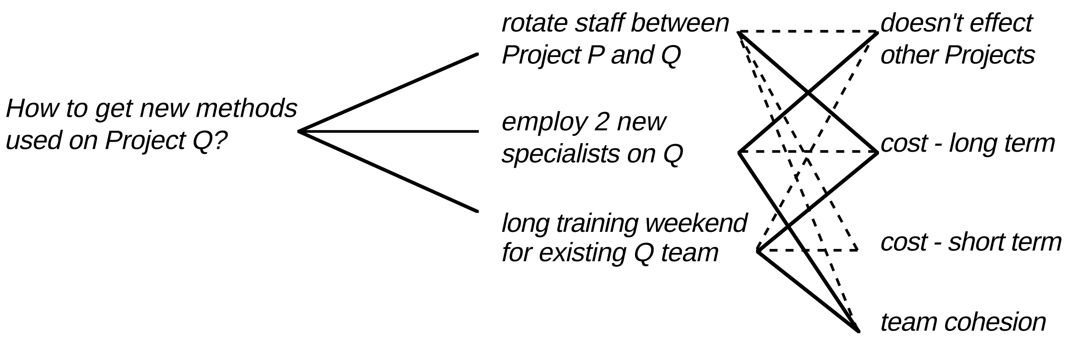
Code level QOC analysis



User interface level QOC analysis



Planning level QOC analysis



Basic conventions in QOC

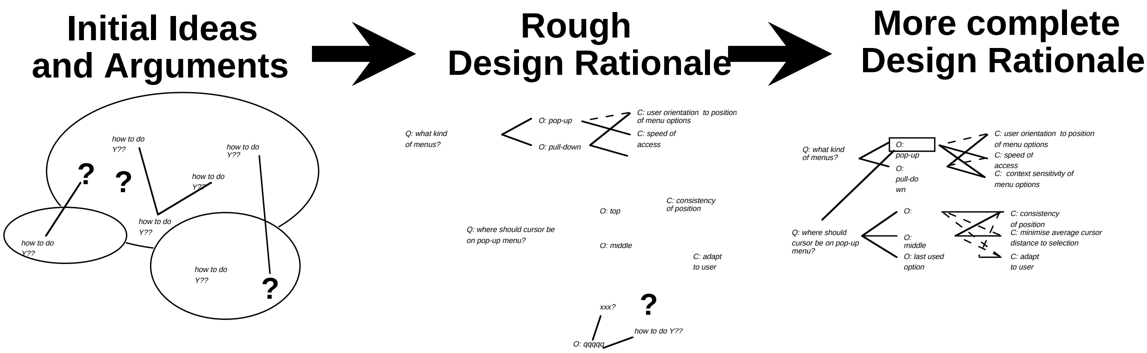
- put a box round the Option which is chosen to show it is a decision
- use solid lines to show *Supports* relationships from a Criteria to Options, and dashed lines to show *ObjectsTo* relationships.
- so that a solid line always indicates a ‘plus’ for an Option, and a dashed line a ‘minus’, Criteria are expressed positively, eg. “simplicity” rather than “complexity”; “ease of use” rather than “difficulty of use”. This avoids confusion over double negatives, eg. a negative link to “hard to learn” is logically positive.
- since in problem solving **it’s impossible to produce beautifully clear solutions from scratch**, it won’t always be immediately obvious what kind of entity an idea or argument is, or how it relates to the rest of the DR. Things become clearer only as the problem is talked through and the issues better understood. This leads us to...

Suggested authoring strategy

- **record reasoning as QOC where possible**, but make rough notes and sketches in a separate workspace, **integrating them into the DR as their status becomes clearer**

- **have all the main arguments recorded at least as "rough DR"** when you come out of a meeting; this record can then be refined subsequently as analysis continues — the main step to creating intelligible DR is to get ideas down in that form initially, and then to work from that – tune it up as and when you spot problems, but DRs often undergo iterations 2 or 3 times, eg. as you realise that a Question has only 1 Option; that nodes have names which only make sense to you; that a Question can be broken down into two sub-Questions; that 3 Criteria in different places are essentially the same, and can be renamed at a more abstract level to show they're being reused, and so forth.
- **continually appraise your DR for coherence**; as you get used to this mode of working, the closer your "rough DRs" will be to the "finished DR"

This approach to creating DR is shown schematically below.



The payback for using design rationale

QOC, and DR in general are so new that instances of their use in commercial design environments are thin on the ground. Many of the benefits from using it are reported from its use within research groups on their own projects, and it is not uncommon for the developers of a particular DR notation to experiment with it themselves. Whilst this can be viewed as a limited 'field test', results from such exercises are inevitably somewhat limited in scope, as the 'DR users' are all DR experts, or are perhaps all of a ready disposition to try out new ideas at work. The real data will come from companies exploring DR within their everyday work.

However, below is a summary of benefits reported so far from studies of DR use. It will be interesting to see the extent to which these are corroborated, challenged, and added to over time.

Atlanta NCR field study [Conklin & Yakemovic, 1990]

- given hands-on training (very important), DR learnt quite quickly
- more complete, consistent meetings records (notetaking role rotated round team members)
- improved analysis of problems
- more effective meeting structure
- quicker reference to previous decisions
- project management (esp. with tool support: tracking unresolved issues; action items raised by decisions)

Cambridge research [MacLean, 1989/1991]

- QOC shows how alternative designs relate to each other within the design space
- there is fairly high compatibility between the vocabulary of designers' discussions (without DR) and QOC concepts (but asking good Questions needs working on)

York research [Shum, 1991]

- DR acts as an external memory to remind designers of unresolved points
- graphical form highlights incomplete reasoning, particularly in evaluating Options against all relevant Criteria
- having to name QOC objects encourages clarity as to the real point being made
- planning the organisation of your issues (what Questions to ask) facilitates problem analysis

Summary

Generating design rationale should (i) be helpful to you as designers, and (ii) convey to others the reasoning behind your final design. More specifically:

i) organising your ideas using this notation should help you think more thoroughly as to why exactly you make the decisions you do;

ii) design rationale should be organised so that an outsider can understand the issues underlying a design, the space of alternative paths for each, and reasons why one rather than another was chosen. Coherent rationale may emerge by just writing down Questions, Options and Criteria as you think of them, and as you become more familiar this will occur more often. However, a whole DR can rarely be constructed on a single pass – it is necessary to reflect on the best way to organise and name QOC nodes, and change parts of the original DR in order to make the whole clearer.

Rather like a textbook example, instead of a literal record of the misunderstandings, false starts etc. that were made, the final design rationale is an *idealised, crafted account* of how you structured and solved the design problem. The aim is to convey as simply and effectively as possible, the 'space' of possibilities in which the design sits.

Additional reading (available from Department of Psychology, University of York)

Conklin J. and Begeman M.L. (1989) *gIBIS: A tool for all reasons.* Journal of the American Society for Information Science, May 1989, 200-213

Fischer G., McCall R. and Morch A. (1989) *JANUS: Integrating hypertext with a knowledge-based design environment.* Proceedings Hypertext'89, Pittsburgh, 105-117. ACM: New York

Lee J. (1990) *SIBYL: A tool for managing group decision rationale.* Proceedings Computer-Supported Cooperative Work (CSCW'90), Los Angeles. ACM: New York

MacLean A., Young R., Bellotti V. and Moran T. (1991) *Questions, Options, and Criteria: Elements of design space analysis.* Human-Computer Interaction, 6 (3+4)

McCall R., Bennet P., d'Oronzio P., Ostwald J., Shipman F. and Wallace N. (1990) *PHIDIAS: A PHI-based design environment integrating CAD graphics into dynamic hypertext.* Proceedings of ECHT'90, 1st European Conference on Hypertext, Paris.

Shum S. (1991) A cognitive analysis of design rationale representation. Ph.D. Thesis, Department of Psychology, University of York, UK

APPENDIX 15: CLASSES OF CRITERIA GENERATED AND USED BY CASE STUDY 3 DESIGNERS

The Impact Project studied in Case Study 3 were refining, and then applying a comprehensive set of criteria to a set of software packages for production planning, in order to make a decision. Below is a selection of the criteria which were agreed on within the project.

Data import/export	***Functional model building***	***User interface***
data formats supported interchange with existing systems interchange with PC packages	ease of model building switchable constraints effective diagnostics simulation of multiple 'what-ifs' run multiple trial models	Common User Access menus and command line graphical presentation speed of response pan and zoom adding, displaying, moving sites context sensitive help quality of manuals flexible macro language ease of manual scheduling
Data structure	***Consolidation of forecast demand***	***Language support***
flexibility and comprehensibility SQL compatibility disk space requirements	read forecast of sales demand by geography and customer read actual orders with dates evaluate appropriateness of promotions, price and weight changes, etc.	multi-lingual capability European keyboard support online help and tutorials printed documentation
Platforms	***Allocation to production***	***Reporting***
runs on existing platforms long-term availability	(allocation of volume to production facilities, over time and across sites, taking into account) multiple-site sources	configurable (by builder and user) printer & plotter support exceptions visibility of information
Availability	***Algorithm support***	***Distribution and outline planning***
total installation cost and maintenance Non-US version promised versions?	range and speed of standard set macro capability of scheduling criteria	storage projection storage consolidation distribution projection consolidation of distribution demand