Why patterns are not enough: some suggestions concerning an organising principle for patterns of UI design

Sally Fincher, Computing Laboratory, University of Kent at Canterbury, UK
Peter Windsor, Technical Director, Usability Limited, 21 Rock Road, Cambridge CB1 7UG [i]

Introduction

Much previous work in the HCI arena has focussed on Capture of Practice, Abstraction and Presentation [ii] – that is to say the writing of individual patterns – generally with the explicit rationale that these are the "building blocks" and an organisation for them will come later.

However, we took the view that a pattern *language* is a gestalt product. Thus, the identification and selection of specific practice *as* patterns may be influenced (perhaps, even, should be influenced) both by the relationships between patterns – or pattern categories – and by the anticipated users (and use) to which the patterns will be put.

Suggestions towards an Organising Principle

We thought that an organising principle (as opposed to an individual pattern) should, as a minimum,

- **Taxonomise** It must allow users to locate or select material from a large corpus; to find the pattern(s) they need.
- **Proximate** It must allow users to locate supporting, perhaps inter-related, patterns applicable to their solution – both "broader" and "narrower". (This use mirrors known designer behaviour, to jump between levels and layers of the problem, undertaking "opportunistic forays into design detail")
- **Evaluative** It would be desirable if an organising principle allowed users to consider the problem from different viewpoints – so that they could evaluate and change their approach, or, equally, confirm the quality of their existing solution
- **Generative** It should allow users to build new solutions, not previously considered.

With this in mind, our first thoughts were to mirror the Alexandrian structure of scale. Just as architects design at different scales, so do UI designers. It was not difficult to devise a similar hierarchy of scale:

SCALE
"Society"
System
Application
UI Structure
Component
Primitive
Physical Detail

Not only did this (accurately enough) reflect the constraints of the medium, but we were confident that there were already patterns in existence that would fit, especially at the "component" and "primitive" levels where (we suspected) we know the most. However, we did not feel that this was, in and of itself, sufficient. Architecture is primarily concerned with the physical world: Popper's "World 1" [Popper quoted by Gaines, Interact 99] . HCI, however, is concerned with Popper's "World 3", the world of knowledge which has a far richer structure. Architects design physical artefacts, and therefore think in terms of three-dimensional space. UI designers think of their artefacts in many different ways, and we were anxious that they should be able to find a pattern for their problem whatever their conceptualisation of the design process.

Our second structure was based on the conception of design-by-type-of-task, where we defined "tasks" by primary reference to the flow of information interaction. Thus:

TASKS	**INFORMATION INTERACTION**

Retrieval	Retrieval tasks have (static) information passing from the artefact to the user(s). The flow is usually initiated by the user(s).
Monitoring	Monitoring tasks have (dynamic) information passing from the artefact to the user(s). The information may come from 'beyond' the artefact. The flow is usually initiated by the artefact.
Controlling	Controlling tasks have information passing from the artefact to the user(s) and a separate flow from the user(s) to the artefact. The flow may be initiated by either the user(s) – proactive control – or by the artefact – reactive control
Construction	Construction tasks have the user(s) putting new information into the artefact
Transaction	Transaction tasks have the user(s) putting linked changes into the artefact. They are often accompanied by a corresponding change in the outside world.
Modification	Modification tasks have user(s) changing information already in the system. They may be modifying 'attribute values' or 'structure'
"Calculation"	Calculation tasks have the user(s) putting information into the system which it then transforms and passes back to the users (not necessarily synchronous).
Workflow	Workflow tasks have the system providing information to the user(s) which they then transforms and passes back to the system (not necessarily synchronous).
Communication	Communication tasks have one group of users putting information into the system that it passes to another group of users.

This was well enough. But there are situations and designers for which this would be a problematic (indeed torturous) way to think. It is as common, as "natural", for UI designers to structure their design not around the nature of the interaction (the "how"), but the stuff that is to be interacted with (the "what"). So, we created a third categorisation:

INFORMATION
Volume
Complexity
Structure:
amorphous
sequential
hierarchical
directed acyclic graph
web
Dynamics:
creation/termination
rate of change
patterns of change

There are many similar examples of taxonomies of/for design space. None of the three we have identified here may be any "better" than any other. However, they all failed to capture the feeling which pervades the Alexandrian work. They were all to do with the activity of design and not with people, or the world, or (especially) real people in the real world. They did not reflect values. Stepping back, we tried to capture categories which would allow us to express this (via the constituent patterns):

CULTURE/SOCIETY
ENVIRONMENT
ROLE
USE
"NAVIGATION" (through SPACE/TIME)

This was clearly of a different order than the taxonomies we had delineated before. It both preceded and pervaded their concerns. For example, it is difficult to think of the design of a retrieval task that would not be substantially changed by consideration of the values of the environment in which it was to be situated – if it were to be placed on a boat then factors of the physical environment (such as vibration and noise levels) would be important, if in a company intranet then factors of the organisational environment (who has access to what and why) would be higher-order considerations.

Whilst this categorisation allowed us to reflect values, it would not necessarily be of use in finding any given pattern. So, finally, we came to the conceptualisation of a two-stage Organising Principle:

CONTEXT & VALUES	
CULTURE/SOCIETY	
ENVIRONMENT	- organisational
	- physical
	- sphere
	- mobility
	- embedded
ROLE	- single/multi user
	- privacy/security
USE	- anticipated use/ambiguity
	- value: social/professional/quality of system
	- reliability
"NAVIGATION" (through SPACE/TIME)	- affordance
	- exploration/safety

STRUCTURE		
TASKS	**SCALE**	**INFORMATION**
Retrieval	"Society"	Volume
Monitoring	System	Complexity
Controlling	Application	Structure:
Construction	UI Structure	amorphous
Transaction	Components	sequential
Modification	Primitives	hierarchical
"Calculation"	Physical Detail	directed acyclic graph
Workflow		web
Communication		Dynamics:
		creation/termination
		rate of change
		patterns of change

By noting that the various taxonomies could be associated with separate phases in the design process we envisaged that they might be used in sequence, from analysis of the context through consideration of the problem to development of a solution.

Context & Values	Analysis Space
Structure: Tasks	Problem Space
Structure: Information	
Structure: Scale	Solution Space

Of course, such a sequence of events in a design process is not a "one-time-pass", but adding consideration of *process* as well as *pattern* and *language* allows a more generative approach to the idea of an organising principle. It is fairly easy to think of a designer "going round the loop" between the two sections several times in the pursuit of a solution, finding many different patterns, and pattern families on the way.

CONTEXT & VALUES		Analysis Space
CULTURE/SOCIETY		
ENVIRONMENT	- organisational	
	- physical	
	- sphere	
	- mobility	
	- embedded	
ROLE	- single/multi user	
	- privacy/security	
USE	- anticipated use/ambiguity	
	- value: social/professional/quality of system	
	- reliability	
"NAVIGATION" (through SPACE/TIME)	- affordance	
	- exploration/safety	

STRUCTURE				
TASKS		INFORMATION	Problem Space	SCALE
Retrieval		Volume		"Society"
Monitoring		Complexity		System
Controlling		Structure:		Application
Construction		amorphous		UI Structure
Transaction		sequential		Components
Modification		hierarchical		Primitives
"Calculation"		directed acyclic graph		Physical Detail
Workflow		web		
Communication		Dynamics:		
		creation/termination		
		rate of change		
		patterns of change		

Solution Space

A gedankenexperiment

To test our putative organisation, we examined how it would work when applied to a real world problem. We chose a part of the design of the operational telephone system for the Swanwick Air Traffic Control Centre. This evaluation was for the four activities we expect to be supported by a pattern language:

- First, we explored whether it could be used to *find* a primary pattern applicable to the problem as stated.
- Second, we considered whether the language would allow a designer to *locate supporting patterns* to complete the solution.
- Thirdly we examined usage of the PL for exploring the *context* in which the problem was situated.
- Finally, we tried to use the PL to find *radical solutions*, not predicated by the design team's experience or the constraints of that experience.

The problem

We used the problem of ATC Operational phones [Cozens 1998] because it was a complex, real-world problem and we were very familiar with the details.

The essential components of the domain are:

a) There are 3 User Roles (Tactical, Planner, Assistant) for each 'sector'.

b) Each role+sector has a fixed set of other 'addresses' they contact by telephone; some are 'internal' to the operations room, some 'external' to it.

c) Making calls is 'incidental' to Users' central task – so there is a need to minimise the cognitive impact and also to keep the interaction fast. (This may not be identified by the designer "up front")

d) User expectation (shaped by the current systems) is that there will be one 'button' for each person they call, and it will be used for both making and answering calls.

e) High level design has already assigned two 'functions' to a fixed size touch panel:
   - make a call
   - receive a call

   plus other telephone operations that we are omitting for simplicity here.

The design is problematised by the fact that the touch panel is too small to accommodate one button for each 'address' and allow all addresses to be visible at the same time.

Finding a primary pattern

The proposed organising principle proved to be flexible and helpful in finding a primary pattern applicable to this problem. We considered that different designers might characterise the problem in different ways:

- One characteristic of the problem is that it is supporting a communication task. With this in mind we would look in the Communication section of the Task taxonomy, with the expectation of finding an 'opening a channel' pattern there.
- An alternative conceptualisation of the problem is that it is essentially about interacting with medium size set of static objects. This would lead us to the corresponding part of the Information taxonomy (perhaps in the Volume section) where we might to find expect 'selection' patterns.
- The problem could also be considered to be a monitoring task. Again this takes us to the Task taxonomy (Monitoring section) where we might expect to find 'interruption / attention getting' patterns.

This exercise demonstrates that the structure is rich enough to support exploration, even though the problem is loosely defined (so the pattern language can be used at very early stages in the design process) and without depending on a particular conceptualisation of the problem or potential solutions.

Finding supporting patterns

The Scale taxonomy naturally supports clusters of related patterns. The primary patterns for this problem are at the "component" level. At the "lower" levels of primitive and physical detail we would expect to find supporting patterns addressing layout, the representation of state and presentation (eg choice of colours vs symbology). We did not find an example of a "higher" level category applicable to this problem; we believe that this is because we don't have a fully populated pattern language, rather than a lack of insight.

It seems from this example that finding supporting patterns is easily achieved – related patterns "come for free" by virtue of the Scale taxonomy. However, in the absence of a substantial set of patterns, we cannot tell whether this property will be diminished by the presentation of an overwhelming number of choices.

Expanding the problem

If evaluation requires alternatives, it is reasonable to think of the generation of alternatives as "going back to the beginning" of the design process. Thus, we used the Context&Values taxonomy to re-cast the context in which the problem is situated, and hence allow informed choices between alternative solutions. For this example we might:

- *Use the Role section.* Taking as a starting point the fact that this system supports interaction between two equal partners, a pattern such as "Whoever gains the benefit does the work" is suggested. In this example, an implementation of such a pattern recommends that it is more important to make answering a call easier than initiating one. So, given alternative solutions, the design team should pick the one that optimises answering a call.

- *Use the Environment section* In the Physical section, an 'Operations Room' category might contain patterns addressing issues concerned with the interleaving of tasks and the treatment of interruptions. For this example (given low priority within context) an implementation solution such as the ability to turn the ring off might be suggested.
- *Use the Values section* For this example the relevant qualities for the human and professional users of the system are low time on task, low memory load and low cognitive load (because this is an ancillary task being performed within a cognitively complex context). Given alternatives, this might suggest that the design team should choose solutions which are more easily ignored.

The organising principle as suggested would seem to support understanding of the context, and may provide material for comparative approaches, evaluating design choices and the justification of a particular solution.

Finding radical solutions

We believe that a good pattern language can be used to break "out of the box" of habitual approaches. Within the approach we have suggested, we believe that this ability may be facilitated by cross-coupling across the taxonomies: a guided serendipity.

For our problem, some putative examples might be:

- Within the Task taxonomy the conjunction between the communication + workflow sections might suggest integrating the making and receiving of telephone calls with the screen representation of the aircraft with which they are associated
- Across the Task and Information taxonomies, the conjunction of the monitoring + appropriate dynamic characteristic from the Dynamics section might suggest separating the functions of making and receiving a call

This usage is hypothetical. Partly this is because the conjunctions were chosen to fit known alternatives, partly because this behaviour has not been observed in any other Pattern Language. Nevertheless we believe this is a promising idea and the definition is rich enough to support such use.

The way forward?

It may be said that creating an organising principle without a population of patterns is an interesting but ultimately sterile exercise. However, we believe that the converse is also true; that the identification and capture of individual stand-alone patterns without a corresponding structure is an activity which misses an essential meaning of pattern language. We believe that work on an organising principle for patterns should proceed alongside work to write patterns themselves.

References

Fincher, Sally: *Analysis of design: An exploration of patterns and pattern languages for pedagogy*. Journal of Computers in Mathematics and Science Teaching: Special Issue CS-ED Research, 18(3):331-348, December 1999.

[i] This paper reports on and presents ideas first discussed during the *Usability Pattern Language Workshop* at INTERACT'99. Participants in these discussions were: Sally Fincher, Ger Koelman, Martin Hitz, Diane Love, Alistair Sutcliffe, Peter Windsor. Any mis-representations of other contributors ideas are the authors' own.

[ii] Following Fincher [Fincher 99] we take there to be five necessary characteristics of patterns and pattern languages: Capture of Practice, Abstraction, Organising Principle, Value System and Presentation