Using pattern languages in participatory design

DEARDEN, Andy <http://orcid.org/0000-0002-5706-5978>, FINLAY, J., ALLGAR, E. and MCMANUS, B.

Published version

DEARDEN, Andy, FINLAY, J., ALLGAR, E. and MCMANUS, B. (2002). Using pattern languages in participatory design. In: Proceedings of the Participatory Design Conference (PDC 2002), Malmö, Sweden, June 23-25, 2002. Palo Alto, CA, CPSR.

Using Pattern Languages in Participatory Design

Andy Dearden
School of Computing & Management Sciences,
Sheffield Hallam University,
Sheffield, S1 1WB, UK
+44 114 225 2916
a.m.dearden@shu.ac.uk

Janet Finlay, Elizabeth Allgar
School of Computing
Leeds Metropolitan University
The Grange, Beckett Park,
Leeds, LS6 3QS, UK
+44 113 283 2600
j.finlay@lmu.ac.uk

Barbara McManus
Department of Computing,
University of Central Lancashire
Preston,
PR1 2HE, UK
+44 1772 893288
bmcmanus@uclan.ac.uk

ABSTRACT
In this paper, we examine the contribution that pattern languages could make to user participation in the design of interactive systems, and we report on our experiences of using pattern languages in this way.

In recent years, there has been a growing interest in the use of patterns and pattern languages in the design of interactive systems. Pattern languages were originally developed by the architect, Christopher Alexander, both as a way of understanding the nature of building designs that promote a 'humane' or living built environment; and as a practical tool to aid in participatory design of buildings.

Our experience suggests that pattern languages do have considerable potential to support participatory design in HCI, but that many pragmatic issues remain to be resolved.

INTRODUCTION
The pattern language concept was originally developed, by the architect Christopher Alexander and his colleagues, both as a theoretical account of the properties of a humane, or 'living', built environment [2, 3, 5] and as a practical tool to aid participatory design processes [1, 4]. Patterns and, to a lesser extent, pattern languages have been widely adopted within software engineering as a form for sharing knowledge about 'good' design solutions between professionals [15], but the approach to patterns adopted in software engineering has ignored the participatory aspects of Alexander's original work.

In recent years, there has been a growing interest in the use of patterns and pattern languages to support human-computer interaction (HCI) design [8, 9, 31]. Much of this work has been inspired by the perceived success of patterns in software engineering. Of course, the parallels between architectural and interaction design, with their common concern for the design of the human environment, are arguably closer than those between architecture and Software Engineering. This may suggest that the benefits of developing pattern languages in HCI may be even greater than in Software Engineering. However, the approach to pattern languages adopted within HCI has followed closely that of software engineering, with the emphasis on sharing knowledge between professionals rather than on processes to support user participation in design. For example, the definition of a pattern language generated at the Interact'99 patterns workshop states: "The goals of an HCI pattern language are to share successful HCI design solutions *among HCI professionals*…" (our emphasis, as quoted in [9, p39]).

In this paper, we report our experiences of developing and evaluating pattern languages as aids to participatory design of web-based systems. From our studies we have identified a number of important issues that require further examination. These issues may also be of interest in other contexts where externally produced design advice is being used within a participatory design process.

Structure of this paper
In the next section, we introduce the concept of patterns and pattern languages as used in architecture, software engineering and HCI. We then describe the approach we are developing for using pattern languages in practice and how it relates to Alexander's approach. We then make a number of observations both about the form of pattern languages and practices using them derived from our investigations. Finally, we discuss relationships with other work, and issues we hope to address in the future.

PATTERNS AND PATTERN LANGUAGES
Pattern Languages in Architecture
Alexander introduces design patterns as follows:

> "Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice" [2, preface p. x].

Alexander's pattern language includes patterns addressing

different physical scales ranging from the distribution of cities [2, pattern 1], the organisation of communal space, e.g. 'Access to Water' and 'Accessible Green' [2, patterns 25 & 60], through to patterns addressing detailed structure in individual rooms, e.g. 'Windows which Open Wide' and 'Alcoves' [2, patterns 236 & 179]. An intermediate level pattern is 'Light on Two Sides of Every Room', for which the problem and solution are stated as:

> "**When they have a choice, people will always gravitate to those rooms which have light on two sides, and leave the rooms which are lit only from one side unused and empty.**
>
> …
>
> Therefore:
>
> **Locate each room so that it has outdoor space outside it on at least two sides, and then place windows in these outdoor walls so that natural light falls into every room from more than one direction.**" (2, pattern 159, authors' emphasis)

For "convenience and clarity" [2, preface, p. *x*], Alexander defined a specific textual and typographical format for the presentation of a pattern, consisting briefly of: a name and reference number; a picture showing an example of an instantiation of the pattern; a paragraph to set the context; three 'diamonds' marking the start of the problem; a concise problem statement (emboldened); the body of the problem, including the empirical background (the motivation for the pattern) and the 'forces' involved in the resolution of the problem; a solution (emboldened and preceded by the word 'Therefore'); a diagram to illustrate the solution; another three 'diamonds' to mark the end of the problem; and a paragraph indicating how this pattern relates to other 'lower' patterns in the pattern language. Important features of this format are:

- the combination within each pattern of both abstract descriptions of the solution (in text and graphics) and an illustration of a concrete realisation of the pattern;
- the inclusion of explicit advice recommending a specific built form, rather than simply stating desirable properties of a 'good' solution;
- the combination of both the problem – solution pair (emboldened) together with text providing a rationale for the particular solution recommended.

Patterns within the language are related in a hierarchy with larger-scale patterns indexing patterns at smaller scales that can be used in their realisation. In [2 & 3] Alexander develops an explicit analogy between the concept of a generative grammar for natural human language and pattern languages in architecture:

> "both ordinary languages and pattern languages are finite combinatory systems which allow us to create an infinite variety of unique combinations, appropriate to different circumstances .." [3, p187].

The parallels between natural languages and pattern languages also relate to the way that Alexander understood the evolution and development of pattern languages. Alexander viewed pattern languages as shared cultural artefacts, reflecting the practices of the communities that developed them. He interpreted the development of design languages by professional communities, in ways that excluded the users of buildings, as part of what he viewed as the failure of modern architecture. One effect of this was that:

> "Specific patterns, like, for instance, the light on two sides pattern, vanish from people's knowledge about building … And those few patterns which do remain within our languages becomes (sic.) degenerate and stupid."[3, p235].

Thus he claims:

> "So long as the people of a society are separated from the language which is being used to shape their buildings, the buildings cannot be alive.
>
> If we want a language which is deep and powerful, we can only have it under conditions where thousands of people are using the same language, exploring it, making it deeper all the time. And this can only happen when the languages are shared."[3, p241, 242].

For Alexander, pattern languages were, in part, a way of sharing knowledge about building throughout a society. The concept of local and culturally specific pattern languages can also be found in his work. For example, King [18] discusses the development of a specific pattern language to support the design of a school in Japan, which draws upon the earlier languages, but is specific to the particular community for whom the building is intended. There are parallels to be drawn between Alexander's description of pattern languages and Ehn & Kyng's [13] discussions of design as a language game, and the concept of speech communities discussed by Wynn & Novick [32].

Design patterns in software engineering

Early in the 1990s many software engineers were seeking ways in which design knowledge could be represented and shared between practitioners [6]. This led to an interest in the works of Christopher Alexander and resulted in early workshops at OOPSLA [11, 7] and then to the Pattern Languages of Programming conference series [12]. Discussed at these conferences are patterns and pattern languages that address many topics including the organisation of software projects and teams, design of user

interaction, and software architectural design.

Perhaps the best known work associated with this series of workshops and conferences is Gamma et al.'s book 'Design Patterns: Elements of Reusable Object Oriented Software' [15]. Gamma et al. state that a pattern has four essential elements, a pattern name, the description of a problem, a solution and a discussion of the consequences, i.e. costs and benefits, of applying the pattern. Examples of object oriented design patterns include 'Observer' (a generalisation of the familiar 'model-view-controller' architecture for user interface construction), and 'Command' (a software design to implement undoability).

Although Gamma et al.'s patterns do contain cross-references to each other, the patterns do not form a generative language. Rather, the authors refer to their collection as a "catalog". Unlike Alexander's pattern language which has a specific starting point (a root node within a graph of patterns), finding a pattern in Gamma et al.'s catalogue assumes an initial search process. Coplien & Schmidt [12] discuss the differences between pattern languages and pattern catalogues in software engineering.

Patterns and Pattern Languages in HCI

HCI has seen examples both of pattern catalogues [16, 29] and of pattern languages [9, 27]. Whereas software engineering patterns generally describe the structure and execution of software, for example identifying classes and messages between objects, HCI patterns describe properties and behaviours of interactive systems that can be perceived by users. For example, one pattern from Tidwell's "common ground" language [27] is 'Progress Indicator' for which the context, problem and solution are stated as:

"**Context:** A time consuming process is going on, the results of which are of interest to the user.

Problem: How can the artifact show its current state to the user, so that the user can best understand what is going on and act on that knowledge?

Solution: Show the user a status display of some kind, indicating how far along the process is in real time. If the expected end time is known, or some other relevant quantity (such as the size of a file being downloaded), the always show what proportion of the process has been finished so far, so the user can estimate how much time is left. If no quantities are known – just that the process may take a while – then simply show some indicator that it's still going on …"

The pattern is illustrated with a picture of a dialogue window, showing a progress indicator for a file transfer.

A natural question for HCI patterns is how they differ from guidelines or heuristics. There is, in one sense, nothing new in patterns [10]. Patterns are an attempt to record principles that are already known to 'good' designers. However, patterns combine abstract statements of design principles with: descriptions of the context where the pattern can be applied; concrete illustrations of how the pattern might be realized; discussions of the rationale for the solution chosen; and examination of relevant trade-offs that may need to be considered. Hence patterns represent a particular choice for a way of communicating design advice, and may be regarded as more closely related to the 'Claims' work of Carroll & Sutcliffe [25, 26] than they are to work on heuristic evaluation or style-guides.

This issue of patterns as a communication medium has been explored by Erickson [14] and Borchers [9]. Borchers discusses the use of pattern languages to support communication between three domains of expertise in developing multimedia exhibits. He presents a pattern language for the production of blues music, a pattern language for designing interaction with multimedia exhibits, and a language addressing software architecture issues relevant to such exhibits. By encouraging each of these separate disciplinary groups to utilize the pattern languages within design discussions, Borchers promotes patterns as a medium to improve communication across disciplinary boundaries. Erickson [14] takes this position a stage further, speculating on patterns as a possible 'lingua franca' (common language) for all design stakeholders. Erickson explicitly recognizes the importance of including users as participants in this conversation. However, Erickson's work is primarily a speculative discussion of how patterns might contribute to such developments, and he explicitly stated that his ideas had not been applied in practice. Martin et al.'s [19] work presenting findings from ethnographic studies of co-operative work can also be understood as an attempt to exploit the pattern form to aid communication between professional disciplines.

A natural question is whether pattern languages can actually advance active user participation in design. We examine this question in the rest of this paper.

DEVELOPING A PROCESS

In this section, we review Alexander's approach to using patterns languages, and describe the approach we have adopted for participatory design of websites.

Alexander's process model

As we have noted, pattern languages in architecture were originally developed as tools to support participatory design. In a series of case-studies, Alexander *et al.* describe the participatory processes that they sought to develop [1,4,5]. Key elements of these processes were:

1. Removal of the separation of roles between designing a building and realizing it on site, which in Alexander's view, made it impossible to ensure that the building was sensitive to local contingencies. Instead, a new role of 'architect

builder' was introduced, responsible for both assisting the users in design and coordinating building activity on site.

2. The architect builder introduced the users to the patterns in order to support localized control of design. The whole user group addressed patterns covering large-scale issues, such as the relative positions of buildings. As the design progressed, sub-groups considered smaller scale details that particularly affected them. The groups or individuals were asked to consider the patterns, criticize and adapt them to their own situations, and to use them to develop their own designs.

3. When developing designs, users were encouraged to use sketches, and to pace and mark out their designs on the ground where building was to take place. This was important to help them visualize the effect their proposals, in the specific context.

4. Within the building process, Alexander sought to use approaches that supported what he called 'gradual stiffening'. This approach sought to avoid the drawbacks of premature commitment in design, by permitting late adaptations to designs.

A process for interaction design

In seeking to apply pattern languages to interaction design we have adapted Alexander's process, combining it with recognized methods from the participatory design traditions in HCI. Our process is as follows.

1. A designer-facilitator works with the user to develop the design. This designer-facilitator role reflects Mumford's view of a facilitator as one who "will assume the role of guide and helper and assist a user design group to move purposefully along the road leading to a successful system" [22, p.263]. Our designer-facilitator was actively involved with the users during paper prototyping asking questions to make the users think and justify their choices. The facilitator is also involved between sessions in developing more detailed prototypes.

2. Phased introduction of patterns, to deal with different scales of the design problem. For example, the user may first be encouraged to consider content issues, followed by general structural and navigation elements, finishing with attention to detailed layout decisions. This sequencing is reflected in the network structure of the pattern languages we have used. In our work to date, we have not considered the issue of designing with multiple user groups.

3. Concrete representations such as storyboards and paper prototyping are used as the primary medium for early design. Users are encouraged to sketch their own ideas, and to make notes about features they would like to include in the design.

4. Iterative development beginning with paper prototypes and sketches, moving through mock-ups of these designs using web authoring tools, towards finished products. This approach mimics Alexander's 'gradual stiffening' and relates well to work in HCI such as Shipman and McCall's [24] notion of 'incremental formalization'.

Using patterns in website design

In order to test whether pattern languages could be used effectively in participatory design of interactive systems, we have developed two pattern languages, each of which deals with a specific class of website.

The first language addresses the design of travel websites. The language was developed by selecting previously published patterns that address the general issue of interactive systems design, and adapting them to reflect the specific functions and needs of a travel website [23]. This language has been used in seven simulated design exercises, in which different users were asked to develop paper prototypes. The users ranged in experience from a retired teacher with no experience of using the web to a trainee web designer. At the start of the session, users were told that following the patterns was not compulsory, and that the illustrations were examples only and not definitive 'best practice'. Design sessions varied between 1 and 2 hours. After each session, users were interviewed to about reactions to the exercise and to the pattern language.

The second language deals with the design of a web-based learning resource. This language addresses pedagogical, as well as interface design issues. The pedagogical patterns examine appropriate active learning activities to include in a learning resource, for example collaborative learning, exploratory learning and learning by doing. The interface and web design patterns address issues of structure, layout, navigation and user actions. This language was used in six simulated design exercises to develop paper prototypes, and in three further extended studies, in which these initial designs were further developed working through iterations of static HTML and then dynamic web designs. All users in this case were lecturers or students, or both, with some experience of web usage but from a range of academic disciplines. An example pattern from the on-line learning language is shown in the appendix.

In both cases, design work using the languages was videotaped to support analysis of the interaction between users, the designer-facilitator and the design artefacts.

Based on a preliminary (informal) analysis of the data from these studies we identified a number of important issues that

require further examination. These issues involve questions of both the form of pattern languages, and processes that utilize such languages in participatory design. In the next section we present our observations on the use of pattern languages in design exercises.

ISSUES ARISING FROM THE STUDY

In practical design activities, a pattern language cannot be viewed solely as an abstract information source. We must recognize that pattern languages are instantiated by specific physical artefacts, and the structure of those artefacts may have a significant effect on design activity.

Wording the language

The writers of patterns in software engineering have long recognized the care that must be taken in producing a pattern. In software engineering, patterns are developed through successive processes of drafting and revision within 'writers workshops'. Meszaros and Doble [20] present a 'pattern language for pattern writing', that offers guidance on clarity of expression. Meszaros & Doble suggest that pattern writers should identify a clear target audience [pattern D1], and then tailor the terminology of the language to that audience [pattern D2], avoiding detailed explanations of terms that will be familiar to this well-defined group. A recognized consequence of this decision is that "The pattern or pattern language may not be understandable to those readers outside the target audience if the terminology is too specialized."[20, p. 557].

We began with patterns developed for a target audience of other interaction designers, and then made modifications. However, our users were far more diverse in background than this. There were substantial differences in the time spent reading and studying each pattern. Some users appeared to look at the illustrations only, others spent about 20 seconds on each pattern, reading mainly the bold text and looking at the diagrams, whereas some spent as much as 90 seconds reading each pattern in detail. Writing clearly for such a diverse audience presents a significant challenge. It is clear that the 'designer-facilitator' has an important role in supporting users, helping them to interpret the patterns, and interpreting users' statements. We should also be aware of a possible bias towards users who are more comfortable with large amounts of text.

Most of our users appeared to understand the patterns. The fact that the design domain (web pages) was familiar to most of our users was perhaps helpful in this respect. However, we did encounter some problems. One of the patterns we used included the word 'frames' in the context of laying out a web page. One user (a trainee web designer) challenged the pattern, arguing against the use of frames. Another user (a lecturer in a non-computing subject) did not recognize the term, and the facilitator had to repair this breakdown by explaining frames as an implementation technique to break up a page into sub-areas.

This problem could be more acute where patterns are used to design systems that are less familiar to users. For example, at the current time, many users will not be familiar with designs and styles for mobile or wearable systems. Writing patterns to support user participation in such design will present a greater challenge.

The layout of individual patterns

In presenting individual patterns we followed the typographic style adopted by Alexander [2] and by Borchers [9]. This style presents a motivating illustration first. In Alexander's language, this motivating illustration is a photograph of some physical space or object that instantiates the pattern. In Borchers's language, each pattern is illustrated either by a photograph of a user interacting with a system, or a screen shot of a system that exhibits the pattern. In our travel website language, each pattern was illustrated by a screen-shot of a web page that illustrated the use of the pattern. As with Borchers's language, our illustrations were the very first element of the pattern following immediately after the title.

In practice, we found that some users made extensive reference to the illustrations, often without referring to the accompanying text. The users' heavy reliance on the illustrations has two potential disadvantages.

Firstly, the illustrations may give rise to derivative designs, which simply copy "solutions" from the illustration. For example, the pattern "Step by Step" was illustrated by a screenshot from RyanAir.com, that used a circle to represent each step of booking a ticket, and most of our users adopted a similar approach. One user even equated the pattern with the example picture, indicating that the ones she found useful were those with the illustration, the "pattern", which she had incorporated into her design.

Secondly, we observed users referring to multiple illustrations from different patterns when developing their designs. This suggests that if an illustration contains elements that are peripheral to the pattern in which it resides, then users might interpret these elements as recommended practice, even though the pattern author might not wish to recommend these particular decisions.

These disadvantages may be exacerbated by the fact that our illustrations were placed in a prominent position in the layout of the patterns. Some users suggested alternative layouts. These included: placing the problem and solution first, with the explanatory text appearing later; placing screen shot(s) at the end; and using multiple illustrations.

In the design sessions, users reported that they read the problem and solution text, and looked at the illustrations, but only a few of our users actually read the explanatory text. Even where users had not had the opportunity to read the patterns in advance, they typically spent less than 30 seconds reading the pattern before continuing with the

design exercise, suggesting that they were not reading the explanatory text in depth. One user observed: "The style is …quite wordy and could be put more succinctly" (Study2b, User 1). There is clearly a need to reconsider the depth and wording of patterns as well as the layout.

The form of the language

We have experimented with a variety of different physical forms for the pattern language. In the first instance we presented the patterns on single sided A4 paper. Each pattern was presented on one or two sides of paper, stapled together if necessary. In later experiments, we used double sided paper, protective plastic wallets and a ring binder (with dividers) to organize the language.

It appears to be important to be able to handle each pattern individually. This makes it easier for the designer facilitator to introduce patterns into the design discussion, either individually or in small sets. It also enables the user to browse through patterns that they have already seen to find ideas that they feel are useful. During design, users occasionally make reference to information they have previously seen in a pattern, and can indicate this by pointing to an individual pattern, or to a pile of patterns.

During design sessions, we noticed that users progressively handled the patterns more and more, occasionally placing patterns that they had used in a pile away from the designer-facilitator's seat. This may suggest an expression of 'ownership' of patterns, which would be a positive indication user participation.

Our results suggest that the physical affordances of the language are significant for participatory design and that, consequently, efforts to organize pattern languages in hypertext may lose important qualities.

USING PATTERN LANGUAGES

Handling the language

Our results indicate that the behaviour of the facilitator is critical to the effective use of the language. Without exception, users felt that the involvement of the facilitator was vital, the following comment being typical: "at first there was a lot of information and it was important to have you there for guidance and reassurance" (Study 2a, User 1). However, as the sessions progressed, the users were more able to navigate through the language and select patterns themselves. This allows the locus of control over the session gradually to shift from facilitator to user.

The results from our first study suggested that an effective approach is for a small number of patterns (typically between one and four) to be presented together. Users are able to read the problems and solutions quickly before continuing with design. This practice can be used to help the user focus on a small number of relevant usability issues, whilst developing or reviewing some part of the design. We adopted this approach consistently for our second study. We also found it helpful to verify the user's understanding after each set of patterns was introduced, asking questions such as 'what does that pattern suggest to you?'.

In recommending this practice, we should include the proviso that the facilitator must be responsive to user interests. For example, during one session a designer-facilitator is heard to say "*you're jumping ahead, you're good at this*" (Study 1, to User 1) whilst looking for a pattern that was appropriate to the users current focus. In another session, the user indicates that they want their students to examine a series of alternative presentation styles in order. In response, the facilitator suggests looking at a group of patterns that deal with 'step-by-step' instructions (Study 2a, User 6).

The set of patterns can also be used as a "checklist", to ensure that all the issues have been discussed. This can occur in two different ways. Either the list of patterns can be used at the end of a session to check whether all issues have been discussed, and / or the facilitator can use the list to monitor progress, noting when each pattern is used, and constantly reflecting on which pattern to introduce next.

In comparing the designs produced by users, we found that where the patterns were not explicitly managed and presented by the facilitator to the user, certain issues were overlooked. For example, one pattern for travel websites recommends providing feedback about delays that occur when queries are processed. This issue was only considered when the facilitator specifically introduced the pattern. The same result occurred for the idea of including links to other useful sites (e.g. car-hire & hotel booking).

Breakdowns and repair

During the design sessions, breakdowns in communication occurred on many occasions. In such situations, the facilitator is required to identify and repair the breakdown. We observed such breakdowns at three different levels.

At the level of the pattern language artefact, breakdowns may occur where the user misinterprets the intention of a pattern, or of the language. For example, one user reported that when she was told about the hierarchical organization of the patterns, she became concerned that this was a direction to make her website design hierarchical. Another user became confused about the intent of a pattern: "I'm not really sure what it is advising me to do" (Study 2a, User 1). Often the facilitator can avert such breakdowns by discussing ideas from patterns as they arise. Users should feel able to challenge the advice contained in a pattern. Alexander also encouraged this type of dialogue [4].

A second level of breakdown concerns the organization of the design process. Users may be familiar with other design practices such as brainstorming, use of checklists, or spending time studying a large selection of examples before

beginning to produce design ideas. Facilitators need to be aware that users may have previous experiences of design processes that will influence their expectations of the design activity. These expectations can be a source of breakdowns in the design process, and our use of patterns to support participation must itself be negotiable.

Finally, breakdowns can occur at the level of the domain. Our first pattern language was intended to support the design of 'travel' websites. Most of the examples used in the language were drawn from rail and air travel sites (e.g. totaljourney.com, theTrainLine.com, RyanAir.com, EasyJet.com and SingaporeAirlines.com). In one design session, the user interprets 'travel' in terms of package holidays. During the design session she uses the phrase 'holiday site', requests options to select 'hotel or self-catering', and wants to see information on 'transfer time' from the airport to her hotel. These concerns are not well represented by the language, and the facilitator did not recognize this divergence of interests.

These events illustrate the important role of the facilitator in monitoring the progress of the design session for possible breakdowns, and repairing breakdowns when they occur. Whilst breakdowns and repairs are a natural part of any participatory design process, it may be that the use of a pattern language (or any other external advisory artefact) introduces new potential sources of confusion.

The authority of external design advice

The pattern language embeds design advice in a form that is separable from the facilitator, contrasting with the more typical situation in participatory design where advice is offered verbally by a single named individual. This externalization can have a variety of consequences. On the one hand, users may feel more able to challenge the advice offered, since they do not perceive such challenges as a direct conflict with an individual facilitator. On the other hand, users may perceive written information as carrying greater weight than an individual's comments. The behaviour and statements of the facilitator in respect of the language may have an important effect on this balance.

In our design sessions, we tried to present the pattern language as an advisory tool that the user was free to make use of but that required interpretation to the user's specific circumstances. However, as design progressed, facilitators made significant statements that indicate alternative levels of authority should be accorded to the language. For example, we have observed facilitators making the following statements when introducing particular patterns in different situations: "… you might want to have a look at some of these things …"; (Study 1, to User 1)"we don't have to bother about 'language of site' … would you just want it in say English…" (Study 1, to User 7); and "these patterns, they're very much based on … grounded on … usability research ... so what they're actually sort of saying in them has been found through research …"[Study 2a, to User 5]. Such statements may significantly alter a user's attitude to the information presented in the language.

Certainly some users expressed their "trust" in the patterns, and indicated that they were happy with their designs because the patterns were "correct" (Study 2a, Users 5, 3). This was an unintentional consequence but one which has important implications. The issue of how to introduce materials and practices at the start of participatory design sessions is recognized in the literature [21], but our results show that facilitators might influence users attitudes to patterns throughout design sessions.

Alexander's work also highlights issues of the authority associated with patterns. The patterns in [2] are each rated with a number of stars reflecting the authors' confidence in the correctness and universality of the pattern. In [4] Alexander reports on a conflict in which the users did not agree with a pattern (entrance transition) that he regarded as fundamental. In this case, Alexander insisted that the pattern was adopted in the design but did so without disadvantaging the users (no family had to sacrifice any of their own choices in order to have this feature). In the end, all users agreed that the feature enhanced their homes. This is an interesting example of the resolution of conflict between user and designer. In this case the authority of the pattern was high and therefore was adopted, even though users could not immediately see the benefit. We need to consider how patterns are validated, and how their 'authority' might be mediated, as well as developing our practice in encouraging users to challenge and interpret the pattern within their own context.

DISCUSSION

In our research, we are investigating ways of using HCI patterns within participatory design, an approach that we view as consistent with Alexander's original writings. Our first investigation dealt with an artificial problem, developing paper prototypes for a travel website. On the basis of that initial investigation, we have developed the approach and applied it, with students and lecturers, to the design of on-line learning resources.

Our results to date, suggest that pattern languages might indeed be useful to support participatory design activities. Overall, users responses were positive, and, once they became familiar with the use of the patterns, they reported that they found the patterns helpful. Of course, we must question the extent to which these positive responses can be attributed to the use of the pattern language, as opposed to the experience of paper prototyping or factors relating to the facilitator.

Related work

The majority of previous work on pattern languages in HCI has focused on the problem of identifying and documenting

patterns. See [9, 16, 19, 27, 29]. In our work we have explicitly sought to avoid writing new patterns, preferring to investigate the problems of applying patterns in practice. Other researchers are also beginning to investigate these issues [31]. This practical focus places our work in close relationship to work on 'Tools for Working with Guidelines' [28]. van Welie et al. [30] suggest that patterns could be superior to guidelines as tools to support design practice, but do not provide detailed evidence. Henninger [17] discusses the application of patterns to multiple projects in an organizational learning framework. However he does not examine participatory design, or present analysis of the processes of applying patterns within design. We are not aware of any other work, to date, investigating HCI pattern languages as aids to user participation.

Further work

If we are to realise the full potential of pattern languages to support active user participation in design, our work raises questions of both the pattern language form and facilitation methods that require further work.

With regard to the form of patterns, we are concerned about wording, layout and physical affordances of pattern languages. Our results suggest that users did not find the Alexandrian layout particularly accessible. We are exploring different formats, including "cut down" versions and different orderings of text and illustration.

We also need to refine the facilitation process to enable users to understand the process and the pattern language and to negotiate solutions suited to their own contexts. Alexander viewed pattern languages as fluid and evolving through use. In our studies we saw how this might happen through negotiation and discussion with users. However, our evidence also suggests that some users rely (heavily) on the patterns as authoritative guidance. We need to examine ways of validating patterns, and facilitation practices that emphasise interpretation of patterns in the local context, and the possibility of challenging patterns.

Our ongoing work is to investigate these issues with more users and over longer time frames. We are revising the on-line learning language and will develop and use it with a broader range of educators and students in real design activities. We are also engaged in the development of a medical portal website in collaboration with a group of users. We are evaluating a range of pattern language formats and hope to apply pattern languages in more realistic scenarios involving groups of stakeholders, rather than small numbers of individuals.

Finally, we need to investigate the issue of the quality of the outcomes. Alexander was seeking the "Quality without a Name" [3]. Both the process itself and the products that are developed through it, should contribute to improvements in the quality of life of participants. In our future work we hope to examine whether our pattern languages and processes can help to achieve this aim.

ACKNOWLEDGEMENTS

We acknowledge the support of our respective institutions for this research. We would like to thank the participants in our design studies for their cooperation and Kay Plowman for initiating the Travel Web Site Language.

REFERENCES

1. Alexander, C., Silverstein, M., Angel, S., Ishikawa, S., and Abrams, D., 1975. *The Oregon Experiment*. Oxford University Press, NY, USA.
2. Alexander, C., Ishikawa, S., Silverstein, M.,Jacobson, M., Fiksdahl-King, I., and Angel, S., 1977. *A Pattern Language*. Oxford University Press, NY, USA.
3. Alexander, C., 1979. *The Timeless Way of Building*. Oxford University Press, NY, USA.
4. Alexander, C., with Davis, H., Martinez, J., and Corner, D., 1985. *The Production of Houses*. Oxford University Press, NY, USA.
5. Alexander, C., Neis, H., Anninou, A. and King, I., 1987. *A New Theory of Urban Design*. Oxford University Press, NY, USA.
6. Anderson, B., 1993. Addendum to the Proceedings of OOPSLA '92. Workshop Report: Towards and Architecture Handbook. *OOPS Messenger* 4(2), 109 – 113.
7. Anderson, B., Coad, P and Mayfield, M., 1994. Addendum to the Proceedings of OOPSLA '93. Workshop Report: Patterns: Building Blocks for Object Oriented Architecture. *OOPS Messenger* 4(2), 107 – 109.
8. Bayle, E., Bellamy, R., Casaday, G., Erickson, T., Fincher, S., Grinter, B., Gross, B., Lehder, D., Marmolin, H., Moore, B., Potts, C., Skousen, G. and Thomas, J., 1998. Putting it all together: Towards a pattern language for interaction. *SIGCHI Bulletin*, *30*, 1, 17-33.
9. Borchers, J., 2001. *A Pattern Approach to Interaction Design*. Wiley, Chichester, UK.
10. Cline, M.P., 1996. The Pros and Cons of Adopting and Applying Design Patterns in the Real World. *Commun. ACM* 39(10) 47 – 49.
11. Coad, P. & Mayfield, M., 1993. Addendum to the Proceedings of OOPSLA '92. Workshop Report: Patterns. *OOPS Messenger* 4(2), 93 - 95.
12. Coplien, J. & Schmidt, D., 1995. *Pattern Languages of Program Design*. Addison-Wesley,

Reading MA., USA.

13. Ehn, P. & Kyng, M., 1991. Cardboard Computers: Mocking-it-up or Hands-on the Future, pp. 169 – 196 in *Design at Work: Cooperative design of Computer Sysetms,* Greenbaum & Kyng, Lawrence Erlbaum Associates, Hillsdale NJ., USA.
14. Erickson, T., 2000. Lingua Francas for Design: Sacred Places and Pattern Languages, in *Designing Interactive Systems: processes, practices, methods and techniques* ACM Press, New York, NY, USA. 357-368.
15. Gamma, E., Helm, R., Johnson, R., and Vlissides J., 1995. *Design Patterns: Elements of Reusable Object-Oriented Software.* Addison-Wesley, Reading, MA., USA.
16. Griffiths, R. & Pemberton, L. The Brighton Usability Pattern Collection. http://www.it.bton.ac.uk/cil/usability/patterns/
17. Henninger, S., 2001. An Organizational Learning Method for Applying Usability Guidelines and Patterns. In Little, M.R. and Nigay, L. (Eds.) *Engineering Human-Computer Interaction*. LNCS 2254. Springer, Berlin, Germany 141 – 156.
18. King, I., 1993. Christopher Alexander and Contemporary Architecture. Special issue of *Architecture and Urbanism*, August 1993.
19. Martin, D., Rouncefield, M., Rodden, T., Sommerville, I. and Viller, S., 2001. Finding patterns in the fieldwork, In *Proceedings of ECSCW '01*. Kluwer, Bonn, Germany.
20. Meszaros, G. & Doble, J., 1996. A Pattern Language for Pattern Writing. In Martin, R., Riehle, D. & Buschmann, F. (Eds.) *Pattern Languages of Program Design, 3*, Addison Wesley, Reading MA. USA. 529 – 574.
21. Muller, M. J., 2001. Layered Participatory Analysis: New Developments in the CARD Technique. In *CHI Letters*, 3(1) 90 - 97.
22. Mumford, E., 1993. The Participation of Users in Systems Design. In Schuler, D & Namioka, A. (Eds.) *Participatory Design: Principles and Practices*, Lawrence Erlbaum Associates, Hillsdale, NJ. USA, 257-270.
23. Plowman, K., 2001. A pattern language for travel website design. MSc Dissertation. School of Computing and Management Sciences. Sheffield Hallam University, UK.
24. Shipman, F. & McCall, R., 1999. Supporting Incremental Formalization with the Hyper-Object Substrate. *ACM Transactions on Information Systems,* 17(2), 199-227.
25. Sutcliffe, A G & Carroll, J M, 1999. Designing Claims for Reuse in Interactive Systems Design, *International Journal of Human-Computer Studies*, 50(3), 213-242.
26. Sutcliffe, A G., 2000. On the Effective Use and Reuse of HCI Knowledge*, ACM Transactions on Computer-Human Interaction*, 7(2), 197-221.
27. Tidwell, J., 1999. Common Ground: A Pattern Language for Human-Computer Interface Design. http://www.mit.edu/~jtidwell/common_ground_onefile.html
28. Vanderdonckt, J. and Farenc, C. (Eds.), 2000. *Tools for Working with Guidelines.* Springer, Heidelberg, Germany.
29. van Welie, M., 2001. *Amsterdam Collection of Patterns in User Interface Design*. http://www.welie.com
30. van Welie, M., van der Veer, G.C. and Eliens. A., 2000. Patterns as tools for user interface design. In [28], 313 – 324.
31. van Welie, M., Mullet, K. and McInnery P., 2002. Patterns in Practice. Workshop at CHI 2002. Minneapolis, MN, USA. 21st April 2002.
32. Wynn, E. and Novick, D.G., 1995. Conversational Conventions and Participation in Cross-Functional Design Teams. In Proceedings of COOCS, 95, ACM Press, 250 – 257.

APPENDIX: AN EXAMPLE PATTERN FROM THE ON-LINE LEARNING LANGUAGE

Note: the formatting and typography of this pattern has been adapted from that actually used in our studies for reasons of space and consistency.

CONTROL PANEL (21)

Adapted: Tidwell (1999)

....the user can take actions that affect the existence or state of the whole artifact. Having a control panel it can be used to assist in NAVIGABLE SPACES (16).

❖ ❖ ❖

How should the artifact present these actions?

The user should know exactly how to stop or leave this artifact at any time.

The user should know what other actions are available.

The user may already know what they have to do, but they need to find the corresponding action.

The user may need to perform these in a hurry, or under stress.

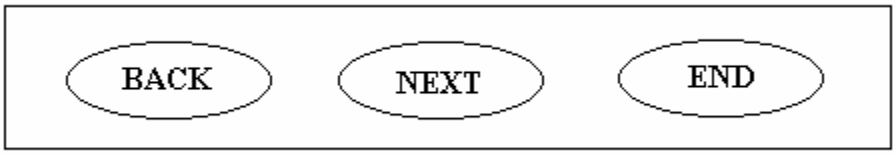
Doing these actions accidentally may be disastrous.

Examples:

- OK / Apply / Cancel buttons on dialogs
- Minimize / Maximize / Quit buttons on Windows application frames

Therefore:

Group these actions together, label them with words or pictures whose meanings are unmistakable, and put them where the user can easily find them regardless of the current state of the artifact. Use their design and location to make them impossible to confuse with anything else.



❖ ❖ ❖

When using a control panel you may need to consider USING COLOUR (30), VISUAL SYMBOLS (27), and USING GRAPHICS (29). You may want to consider SMALL GROUP OF RELATED THINGS (36). When thinking about the controls to use you may want to consider navigation actions such as: CONTINUE TO NEXT STEP (31), GO BACK ONE STEP (33), and GO BACK TO A SAFE PLACE (32).