DONOR PATTERNS: a modular structure for sharing knowledge.

Alan Jessop, David Parker, John Temple.

For publication in the Journal of the Operational Research Society.

Abstract

Groups within organisations learn from experience. The learning experience may itself be enough to help a decision and may also be codified into a knowledge base for use at another time. A modular structure, called a Pattern Language, provides both a base for recording this knowledge and also a format for the debriefing of experience to gain that knowledge. The modularity means that the modules, or prototypes of them, may be imported from other applications so speeding the process and encouraging the transfer of good practices.

Two applications, in process engineering in a chemical company and strategy formation in a small charity, are given which demonstrate the use of donor patterns as means of importing knowledge and stimulating learning. Although the circumstances of the two cases are quite different the processes are similar in duration and structure, suggesting a model for future application.

Keywords: Learning, Decision Support, Practice of OR, Knowledge management, Soft OR

Introduction

There has been for some time a recognition that knowledge has an important role to play in improving the performance of managers, and so of the organisations for which they work.

For knowledge to be shared it must be expressed in a language appropriate to the community of users. There is likely to be a tension between making the language simple enough to promote widespread use and yet sufficiently complex so that the encoded knowledge is not banal. Either too much simplicity or too much complexity may impede use. It is common in these situations to attempt a modular decomposition of the whole, so that each module is sufficiently small to be understood and useful while the set of modules can generate sufficient variety to suit problems as presented.

In this paper a particular modular form, called a Pattern Language (PL), is used to encode experience and so provide a repository which acts as a base for advice on future action. In a PL the modules are called patterns. An example is shown as Figure 1. The components of the pattern (*Context*, *Problem*, *Forces*, *Solution*) provide a structure which acts as a guide for discussion when recording successful actions and also when considering whether to adopt the solution. The pattern contains, in italics, references to other patterns in the PL so that sets of patterns may be assembled as required.

This particular pattern sets out what a pattern is and why patterns are useful. The more general point is that a problem exists when there are opposing forces and that a solution offers some reconciliation.

The two phases – constructing and modifying patterns in the light of experience and then forming advice based upon the patterns – together form a cycle linking knowledge, action and experience. All or just part of this cycle may be used for a given application. The modular form allows the importation of modules from other domains as a way of transferring knowledge directly or as a base for modification before use. This is particularly useful in situations where time is short. Two cases are described where the ability to import knowledge is made possible by the modular approach. These imported modules are called donor patterns.

This paper is organised as follows: some discussion of knowledge usage is followed by a discussion of the usefulness of modular structures and, in particular a PL; two cases are presented which illustrate different applications of the PL method; the main points are discussed and some guidance for use offered.

A. 1 Pattern: Pattern

Context:

You are an experienced practitioner in your field. You have noticed that you keep using a certain solution to a commonly occurring problem. You would like to share your experience with others.

Problem:

How do you share a recurring solution to a problem with others so that it may be reused?

Forces:

- Keeping the solution to yourself doesn't require any effort.
- Sharing the solution verbally helps a few others but won't make a big impact in your field.
- Writing down your understanding of the solution is hard work and requires much reflection on how you solve the problem.
- Transforming your specific solution into a more widely applicable solution is difficult.
- People are unlikely to use a solution if you don't explain the reasons for using it.
- Writing down the solution may compromise your competitive advantage (either personal or corporate).

Solution:

Write down the solution using the pattern form. Capture both the problem and the solution, as well as the reasons why the solution is applicable. Apply *Mandatory Elements Present* to ensure that the necessary information is communicated clearly. Include *Optional Elements When Helpful* to capture any additional information. Distribute the resulting pattern to the largest audience you feel it could help that does not compromise your competitive advantage. Often, this means publishing your patterns exclusively within your company via Intranets or company journals.

Figure 1. A pattern about patterns (Meszaros, 2014)

Knowledge and advice

The performance of organisations is to some extent determined by lessons learned from past decisions and their consequences. This organisational memory has been defined in a number of ways but essentially as "the means by which knowledge from the past exerts influence on present organisational activities" (Stein and Zwass, 1995). This influence is not always benign and may lead to low as well as high levels of performance (Jackson, 2012; Stein and Zwass, 1995). While organisational memory may or may not be beneficial, knowledge is seen as always useful: "Knowledge is broadly defined as credible information that is of potential value to an organisation" (Hult, 2003). Because of this and a desire for improvement organisational memory is usually seen as "the sum of all knowledge assets held by an organization" (Kingston and Macintosh, 2000).

Knowledge has two aspects – tacit and explicit (Nonaka, 1994). Cook and Brown (1999) distinguish between that which is possessed – knowledge – and that which is part of action –

knowing. Similar distinctions are made by Ambrosini and Bowman (2001) who propose "that the expression 'tacit knowledge' be replaced by 'tacit skills', implying 'doing'" and by Dyer and Nobeoka (2000) who differentiate between information, which is easily codified and transmitted, and know-how, which is tacit, complex and hard to codify. Shared organisational knowledge is articulated in a set of routines which are revised in the light of new experience (Nelson and Winter, 1982) to develop people's capacity to enact "useful practices" (Orlikowski, 2002).

Knowledge may be transferred so that "useful information or practices" may either be replicated or adapted (Williams, 2007) for immediate or future use. Darr and Kurtzberg (2000) more strictly argue that "transfer has occurred [only] when a contributor shares knowledge that is *used* by an adopter". So there are two motivations: immediate use and more general knowledge enhancement. Transfer is made more difficult as the complexity of knowledge increases. Zander and Kogut (1995) distinguish between the degree to which knowledge is "codifiable" and the degree to which it is "teachable". In the first, users do not have to understand the basis of the transferred knowledge while the second reflects the degree to which the knowledge may be taught, which would involve some understanding. More generally, the speed and success of transfer will be determined by the tacitness, complexity and ambiguity of the knowledge to be transferred (Easterby-Smith et al., 2008) and so the "stickiness" of the process (Szulanski, 2000). These are typical of difficulties which have been familiar to OR workers for some time as "messes" (Ackoff, 1979).

Transfer of knowledge which is both ambiguous and context dependent (Williams, 2007) is made easier if both partners have a shared context for the task (Darr and Kurtzberg, 2000).

The more complex the knowledge the more abstract must be its description and so the more circumscribed the advice based on it. Three levels are shown in Figure 2 in which experience is represented as a number of cases, histories of earlier problems and solutions. Advice for small problems is given by recollection or customary practice, almost unconsciously. At the other extreme, when there are many previous cases, a theory is developed, but only if the cases have sufficient structure to allow analysis. An intermediate class of problem exists in which there are a large number of cases on which advice may be based but they are not, perhaps cannot be, in a form to permit theory building.

Selecting knowledge for use means using analogy to find a match between what is known and what is needed (Rips, 1989). Case Based Reasoning (CBR) (Schank, 1982) retains a large data bank (the intermediate situation in Figure 2) and retrieves relevant cases using a similarity metric and a matching algorithm. This certainly preserves variety. Analogies are made only at the point of retrieval for a particular problem but keeping the case descriptions as full as possible means that "The case-based reasoning paradigm has a bias against the

problem decomposition and recomposition …" and so is used for "barely decomposable" problems (Kolodner, 1993: 16).

Other systems, including PL, encode descriptions of generic problems based on cases rather than use individual cases themselves. Some richness is lost but the processing needed to retrieve knowledge (not cases) is much reduced, probably not needing any computation for retrieval. This more human-centred approach makes high demands of users so there is more scope for biased selection using poor heuristics (Tversky and Kahneman, 1974). This risk is inevitable. The tension between retaining more detail than can be easily (comprehensibly) handled and relying on the assumed skill of a human agent is unavoidable. We opt for the human agent but recognise the need for care.

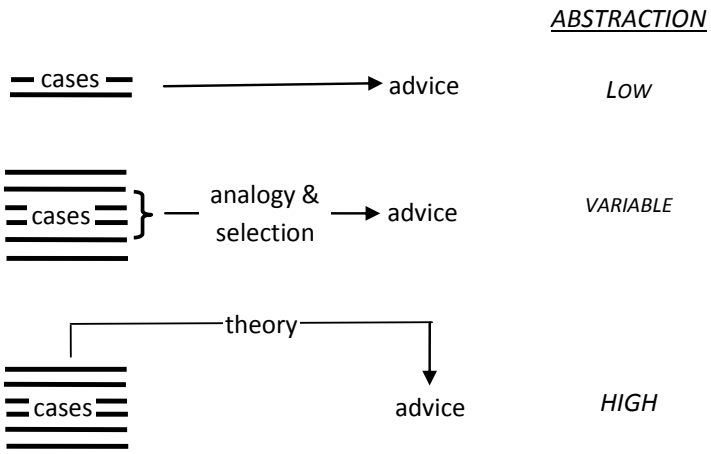

Figure 2. Levels of abstraction

Tasks performed infrequently do not allow the incorporation of the knowledge of past cases, for there is no repetition, and so useful information must be transferred as the basis for *advice* for a decision which may be unique, if not for the organisation at least for those involved. The *context* on which the advice is based must be carefully described and understood so the adopters may assess relevance and potential modification. This problem is particularly acute for project teams (Schindler and Eppler, 2003) and for virtual teams (Kingston and Macintosh, 2000).

Lack of appreciation of context inhibits knowledge transfer. Nevo and Wand (2005) distinguish high context and low context communication. In the first, much information is embedded in the context whereas in the second this contextual information can be inherently part of the knowledge. Ahn et al. (2005) see that "Knowledge is created in various contexts and cannot be perfectly understood when isolated from contexts" which are necessarily part of organisational memory. Just how to describe context is not necessarily straightforward. For

example, Kingston and Macintosh (2000) use multi-perspective modelling though group discussion is an alternative.

Codification of knowledge, particularly tacit knowledge, is usually seen as a prerequisite to transfer (Hall, 2006). Complexity resulting from context dependency and ambiguity argues for a coded form sufficiently flexible to accommodate these difficult and important components. We call such coded knowledge a *pattern*. As Figure 1 shows the patterns used elsewhere (discussed below) have just these characteristics.

Patterns may be used within a group of practitioners as a means of monitoring and encoding evolving practices or they may be imported from another group or organisation, in which case we call them *donor patterns*. Figure 3 shows the process.

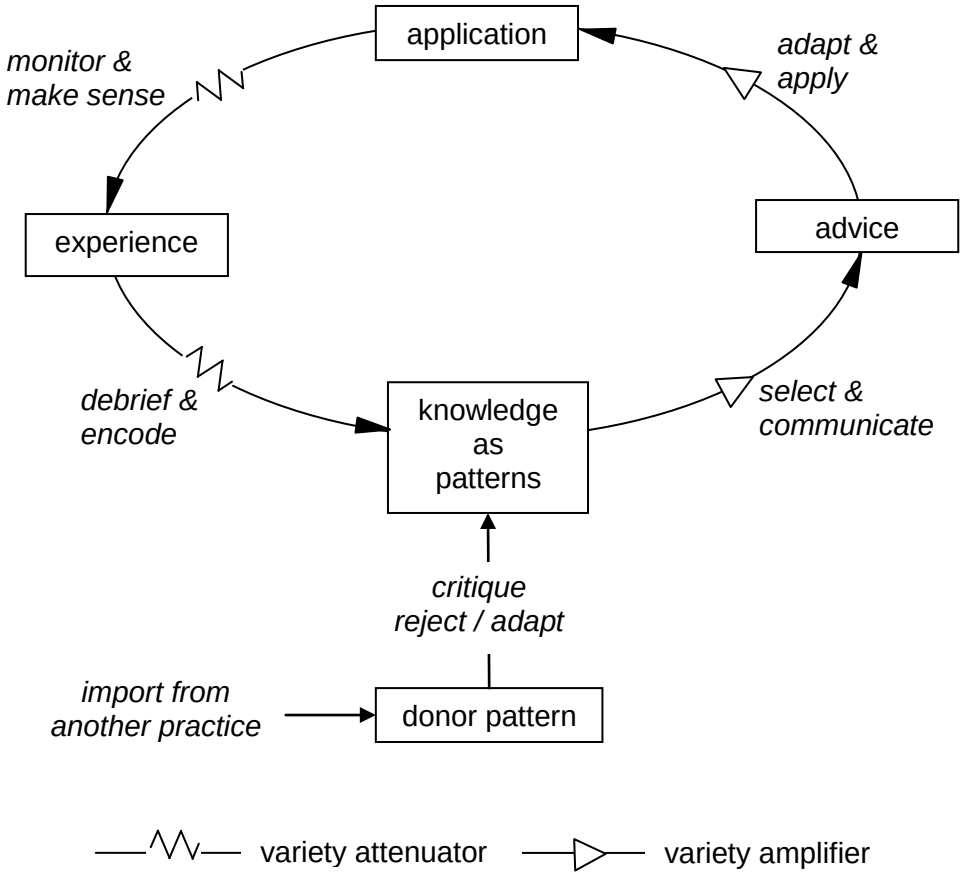

Figure 3. Knowledge sharing using donor patterns.

The main circular path shows how knowledge may be encoded and used. This structure is familiar, appearing in the well-known model of Kolb and Fry (1975) and elsewhere (for instance, in Aamdot and Plaza, 1994). Two influences are at work: the management of variety and the context for action, and so following Beer (1985) the diagram shows how variety changes from stage to stage.

There are two phases. First, a problematic situation as it exists is a high variety complex phenomenon. Encoding experience into knowledge (patterns) is a process of mediated variety reduction. Making sense of what has happened requires discussion in some sort of debriefing (Schindler and Eppler, 2003). This essentially social construction benefits from a *mediator* to help the discussion and generally to manage the process. This phase is shown as the left-hand part of Figure 3.

Second, applying knowledge to construct *advice* on what to do (the right-hand half of Figure 3) needs the skills of an *advisor*; appreciating the nature of the problem, selecting the appropriate knowledge and *crafting* advice. The advisor may also help in application, crafting action from advice by using skills in improvisation (Hatch, 1999; Kamoche et al., 2003; Weick, 1998). Advisor is a role, not necessarily a person, and so may be performed by one of the team.

These two phases, codification and crafting for application, are complementary; codified knowledge supports breadth of use whereas tacit knowledge leads to efficiency of use (Edmondson et al., 2003).

Pattern Language

The use of a PL was initially proposed by the architect and mathematician Christopher Alexander (Alexander et al., 1977) as a way of providing information on which clients could make design decisions either unaided or, more likely, with the help of an advisor (a new role for the architect). Alexander's earlier attempt to find a formal (mathematical) decomposition of design problems (Alexander, 1964) led to the realisation that it was exactly the imperfections in the decomposition which described the richness in the built environment which is so valued (Alexander, 1966). The PL was his response.

Alexander used a discursive pattern form using prose, sketches and photographs. The layout is shown in Figure 4. Emphasis is given by the use of bold type and *** separators.

Alexander's original was taken up most enthusiastically by the software community (Coplien and Schmidt, 1995; Gabriel, 1996; Gamma et al., 1994) in the belief that knowledge sharing would improve productivity by encouraging code reuse. Using patterns for the modular disaggregation of design problems was the same strategy used to write software using routines and functions. The form of patterns changed. Just as the use of photographs and sketches came naturally to architects so a structured form (e.g. Figure 1) was the work habit of programmers.

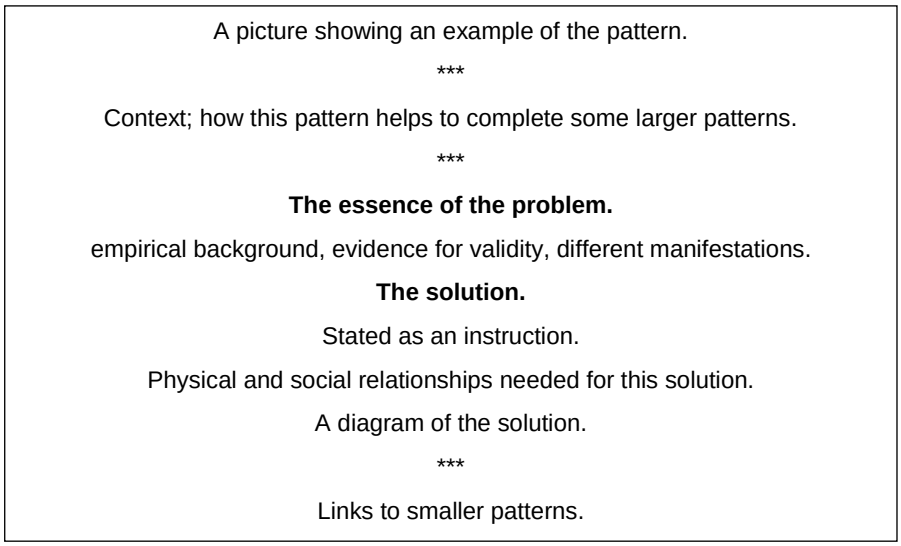

Figure 4. The form of an Alexandrian pattern. (Alexander et al., 1977: x-xi)

Subsequent developments have used many different pattern forms. Some are very brief, Cunningham (1996) has just a two level problem/solution structure, and some much longer resembling a short chapter with as many as fifteen sections (Harrison et al., 2000). Buschmann and Meunier (1995) and Buschmann et al. (1996) suggest that a pattern should have at least the three elements *context/problem/ solution* but may contain more (Figure 5).

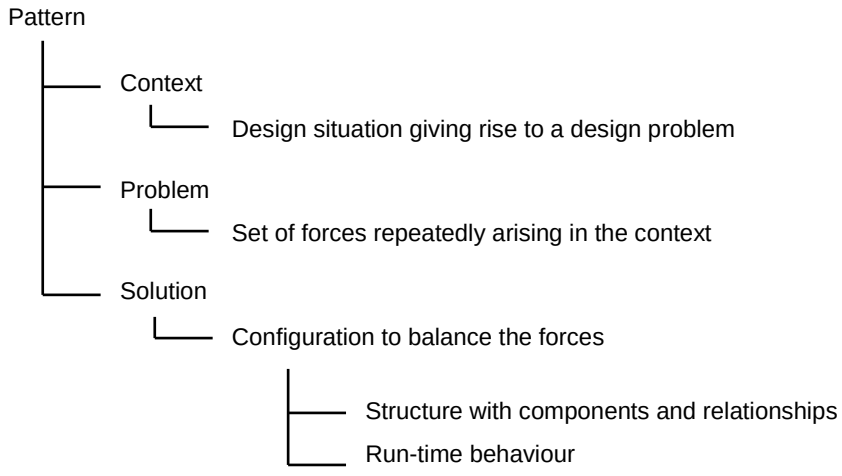

Figure 5. Possible components of a pattern (Buschmann et al., 1996: 11).

The wording shows the origin in software development but is of general applicability: "performance in use" rather than "run-time behaviour", for instance.

The number of categories is likely to be related to the purpose of the pattern. Buschmann et al. (1996: 363) distinguish three:

Architectural Patterns specify the fundamental structure of an application

Design patterns refine and extend the fundamental architecture

Idioms specify implementation, by giving some computer code, for example

It is likely that design patterns will use more categories than architectural patterns and idioms (application patterns) more again. The patterns in a PL may therefore have different forms appropriate to their different functions.

In this paper the structure of Coplien (1995) is used (Figure 6). This form, or something very like it, is one of the most popular. It is simple enough to be readily understood yet has enough detail to be usefully comprehensive. Depending on the subject of the pattern not all of these may be present. For example, in a short technical pattern *discussion* may be unnecessary. Also, nomenclature varies; *rationale* and *discussion* are used interchangeably.

context	describes why and where the problem arises
forces	shows which conflict(s) are responsible for the problem and are to be reconciled
discussion	sets out thinking about the problem and possible solutions
solution	gives the recommended course of action
new context	this advice is likely to solve much of the problem but will necessarily leave a situation which has new and, it is hoped, more tractable difficulties
see also	gives links to other patterns which may also be of interest depending on the prior knowledge of the advisor

Figure 6. Pattern form (after Coplien, 1995).

It is possible that the more structured form will provide good prompts for eliciting patterns but the more discursive presentation may help dissemination to and use by others and so the Alexandrian form has been used by, for instance, the Pedagogical Patterns Editorial Board (2012) and Coplien and Wolfe (2000). A similar pattern format is used by Rogers and Salustri (2009). To see the difference between the two presentational styles compare Coplien (1995) and Coplien and Harrison (2005) in which the same material is presented using both forms. A review is given in Dearden and Finlay (2006). Meszaros and Doble (1998) and the Hillside Group (2014a) give a Pattern Language for writing patterns.

While the overwhelming majority of patterns are concerned with software production other topics have been addressed: Slywotzky et al. (1999) give thirty patterns for business improvement; Coplien and Harrison (2005) write on organisational issues in the context of software development; Manns and Rising (2005) give a PL for the introduction of new ideas. These authors are from the software community and so their approach reflects their backgrounds and interests. This does not mean that what they have written has no wider relevance – building a team is a common task whether it is a team of programmers or a team of chemical engineers and the pattern form, by articulating context and argument, should help knowledge transfer.

Modularity, debriefing and advice

Modular decomposition helps the management of complexity in, for instance, design (Papalambros and Wilde, 2000; Baldwin and Clark, 2000; Hoetker, 2006; Sanchez and Mahoney, 1996; Brusoni and Prencipe, 2006), task partitioning and sharing (Benkler, 2002; von Hippel, 1994), customization (Voss & Hsuan, 2009) and for structuring judgemental decisions (Arkes at al, 2010).

It is the ideal that each module has an independent internal structure linked to the rest of the system via the inputs and outputs specified in the module interface, as in object-oriented analysis. Conceptual models of the system as a whole depend only on interfaces (Fowler, 1997) so that we may switch between the high-level abstraction of the conceptual model and low-level procedural detail of individual modules. This not only helps in system design but also in communicating the behaviour of the system to nonspecialists (Coad and Yourdon, 1991) thereby facilitating discussions which are coherent and meaningful at different levels for different participants. This use of modularity to switch between levels of granularity is just a large-scale formalisation of the chunking strategy used to overcome human information processing limits (Miller, 1956).

A Pattern Language is a modular design for knowledge with benefits both in debriefing and advice-giving.

There are many ways of debriefing to encode practice into knowledge, though this useful task is too often disregarded (Schindler and Eppler, 2003). Using the pattern structure provides an agenda for discussion and a supportive script for the moderator. The modularity enables a free discussion to develop both at procedural and conceptual levels. It is natural to talk about the particularities of a task, a discussion about procedure, while also referring to other modules at the level of conceptual description. Either these other modules already exist or the conceptual references act as markers for their creation. The role of the moderator is to manage this so that no patterns become so large as to be unwieldy (our patterns are usually no more than one or two sides of print) while ensuring that no conceptual links are missed.

Crafting advice from a PL essentially involves the selection of a subset of patterns (Salingaros, 2000) suitable for the current problem. In recognition of the ambiguities which often surround the codification of knowledge each pattern contains an argument to justify a recommendation and a description of context. This allows flexibility in the selection of relevant patterns and permits a number of articulations, determined by the craft of the advisor.

In this paper we are concerned with the use of donor patterns and apply them to different parts of the advice-giving cycle. First, as an aid to debriefing and, second, as a method of transferring knowledge to a group of novice adopters. In both cases assumed names are used at the request of the real organisations with which we worked.

Case 1. Debriefing

The Knowledge Company (TKC, an assumed name) is the research and development arm of a business unit of a large chemicals manufacturer and is concerned with, among other things, the design and improvement of chemical processes for organic acid, the precursor to polymer resin. For the previous decade a progression of teams had undertaken development programmes, valued in total in tens of millions of dollars. These efforts had aimed to improve the variable and capital cost effectiveness of production technology by taking basic physical and chemical ideas through to designs capable of detailed mechanical engineering, construction and operation. These teams had typically involved a core of less than thirty people, working in close proximity and with challenging performance targets. Projects had been scheduled to produce the required output to coincide with projected cyclical market opportunities. The teams consisted of people with experience in the functional disciplines of chemical engineering, chemistry and mechanical engineering who also had experience in a number of the more general management functions. Having been through two cycles of development (call them Dev1 and Dev2) based on conventional technology, the organisation was at the early stage of investigating a new process, Dev3. It was natural to wish to consolidate the learning from the first two cycles before embarking on the third and donor patterns were used for this purpose.

The time scale was short and the amount of time which participants could devote to the exercise was also limited. Donor patterns were used to kick-start the process.

An introductory presentation included example patterns from the language of 42 patterns developed by Coplien (1995) at AT&T Bell Labs. Members of the TKC team who had been through the Dev1 and Dev2 cycles immediately recognised issues and solutions addressed by these exemplar patterns, and so considered it unnecessary to "re-invent the wheel". Using existing patterns in this way is not without dangers, mainly that discussion is biased towards the imported patterns. Given the time pressures for this intervention there were no opportunities to check whether this was happening.

The debriefing group had four members: an engineer and three managers. Although it was originally thought that the team would be most effective if composed entirely of people with experience of Dev1 and Dev2, TKC was concerned that the process be used both as a reviewing and a learning exercise and so a newly appointed technical manager was included. The moderator was himself a former employee of TKC with experience of the earlier projects.

There were six half-day meetings, roughly at weekly intervals. The first three were largely concerned with reviewing the donor patterns; the next two with reviewing the prototype patterns formed by modifying and otherwise editing the donor set; and the final session discussed process and subsequent use. The patterns themselves and the whole idea of the PL were intuitively clear. Some of the donor patterns were seen to be quite readily transferable while others were rejected or heavily modified. The language used in the software business was not always understood. Whether this indicated real differences between the software and chemical engineering environments or just different jargon was resolved by discussion.

The purpose of the project was to explore recurrent experiences in project management. Often, having started talking about the past, the group would then contrast that experience with the current problem. Some previous solutions were thought invalid, but with others the differences were due to a new set of business conditions; the forces and context had changed. This understanding led to patterns based in part on anticipation of the new project as well as on the old. In this way patterns were used as a mechanism to document both experience and intent.

Maintaining discussion at a level appropriate to the task was sometimes difficult. When the scope of a pattern was increasing to an extent that the pattern became too large the moderator intervened to summarise and seek agreement on how to divide the content into other patterns. These patterns were given names and noted. Later in the session they were developed. Switching between the procedural and the conceptual in this way maintained focus without omitting relevant information.

The result of this discovery phase was a set of prototype patterns. The next phase, reviewing, generated rich information and was the point at which the users truly began to own the patterns. In this phase the team began to refer to, and draw operational links between, patterns, and to use the PL terminology more freely. The level of input from the moderator was consequently much reduced. The constant recycling of the discussion until a satisfactory conclusion was reached was important in generating familiarity and ownership.

The result was a PL of 29 patterns. Figure 7 shows an example. The names of all patterns and the linkages between them are shown in Figure 8.

Pattern 14: Organisation Follows Design

Issue: How do you align organisation and process design?

Context: **Organisation Follows Market** [Pattern 12] and **Organisation Follows Location** [Pattern 11] break down the group portfolio into a number of products and associated projects. It is now necessary to shape the breakdown of work within the project.

Forces: The shape of the design and portfolio of products controls the interactions in the organisation. The reverse is also possible.

The shape of the design at the start of the project will change, and the organisation needs to be flexible too.

Solution: At any given stage, let the shape of the product design define the shape of the organisation either as a series of developments or products. Be flexible in enacting a change of roles etc. The leadership team should periodically review the state of the organisation opposite the product needs.

New Context: The design should not be allowed to follow the shape of the organisation: The product portfolio is a variable that can be managed, and so the organisation should be actively adjusted, primarily by ensuring **Role Clarity** [Pattern 31], and **Integrating The Design Teams** [Pattern 35].

Background: The Dev3 project will need an organisational form that accommodates science, commercialisation and flowsheet aspects; this may be a different product-stream orientation to what has been in the past. To date the same structure has integrated all these roles, which has overloaded and diluted the effort.

Figure 7. A Pattern from the TKC case.

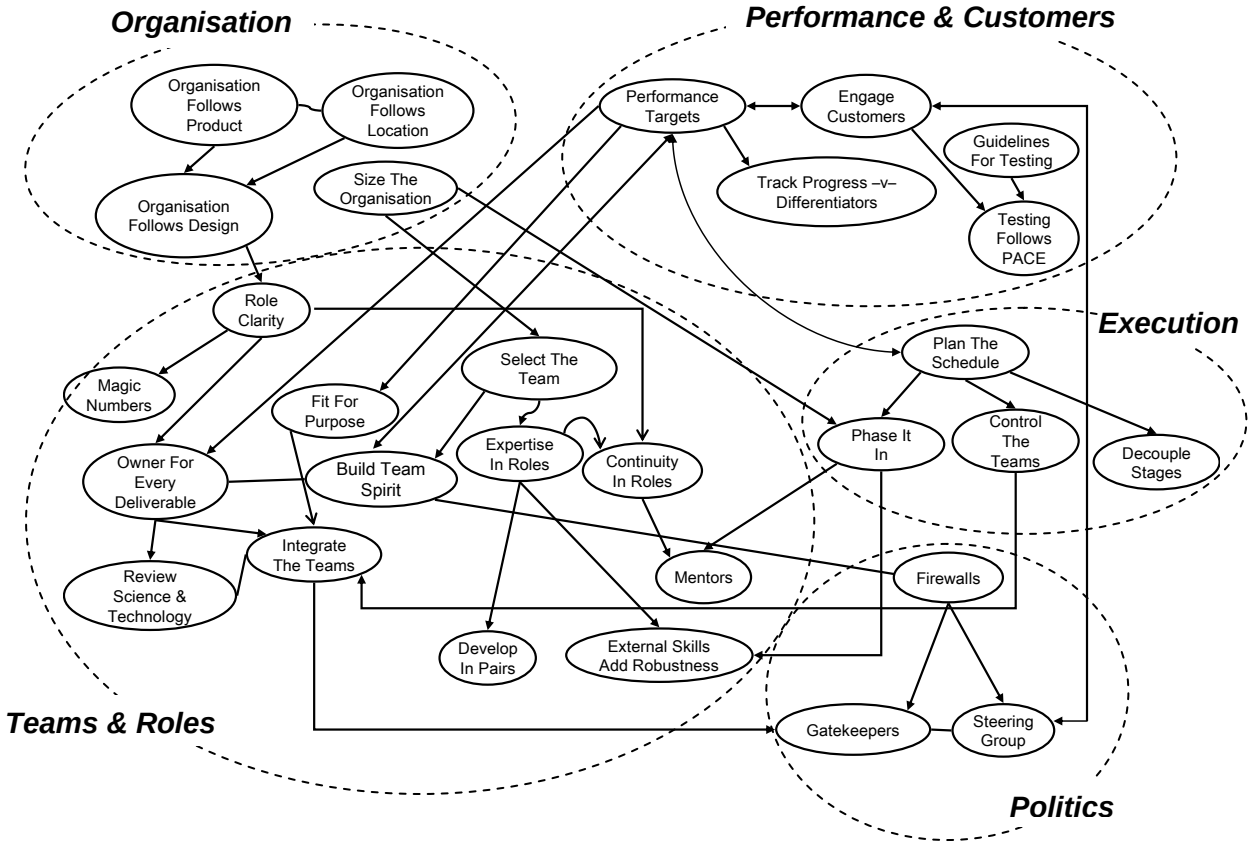

Figure 8. TKC Pattern Map.

How best to present the PL was discussed in the last session which was devoted to issues of use. Finding a presentation of the whole PL in a way that aids navigation to those patterns needed in future applications is an important part of PL design. Maps such as that shown in Figure 8 can help users to identify subsets of patterns needed for a given purpose.

The aggregation into five clusters was meaningful for the group: a useful consensus. This higher level aggregation was part of the debrief and intended to help others in accessing the patterns. However, it may not be unique (another group may make different clusters) and so may not necessarily form part of what is transmitted to others, especially in different organisations.

The Appendix gives transcripts of some of the TKC conversations to illustrate how the donor patterns were used.

Case 2. Learning

The setting for this second intervention was quite different. CF is a charitable foundation situated in 95 acres in the Yorkshire Dales, an area of great beauty popular with walkers and for other outdoor pursuits. CF was established as a conference and holiday centre mainly

staffed by a residential Christian Community. About 5000 guests each year undertake a range of activities which include Church weekends, leisure events (photography, rambling etc.) and small conferences. There are also some private guests. Community members normally join for a minimum of one year. All living expenses are met and an allowance is paid. Community members essentially provide the workforce to run the holiday and conference centre and spend most of their time working on the necessary domestic or administrative tasks and helping to look after the estate.

At the time of the intervention the main governing body was a 15 member Council with ultimate responsibility for running CF including making appointments, creating policy and setting strategy. They also acted as trustees for the charity. A recently appointed Warden ran the day to day business through a small committee. A Management Group had responsibility for practical aspects of planning and helping in the management of the building. A Policy Committee provided a forum for preliminary discussion of policy issues prior to their consideration by the full Council.

The financial position of CF was causing concern. The number of community members had fallen and recruitment was difficult. While plans existed for substantial building works these would have required major funding beyond what was then available. Necessary improvements and remedial work were eating into general reserves.

A Church consultant produced a report which in essence recommended closure or reform, though nothing was said about what those necessary reforms might be. Thinking at a strategic level was not something to which those running CF were accustomed and so there existed a lack of clear direction about the way forward. There was not much time in which to make some difficult decisions. A group of three was formed to take matters forward: the Chair of the Council, the Warden and the Centre Manager. One of us was approached to assist in the process. The role was mainly that of facilitator to help the group arrive at a strategy which they could support.

The situation faced at CF was very different to that at TKC. The issues were strategic rather than operational and the expertise and interests of the people involved had no common technical focus. The need was for problem structuring rather than the codification of existing practice (there was none). As with TKC time was short. The issue here was the presentation of ideas new to the group rather than debriefing based on experience.

It was decided to import some existing ideas which were known to have value of themselves and so would stand for knowledge which, in other situations, the group would be expected to have. The object was to provoke useful discussion, and provide active learning, a consensus view, and a sense of ownership of the result. Based on the experience with TKC it was decided that a PL might provide an appropriate structure. The management literature is not short of advice on strategy and from it the model of Hamel (2000) was thought most

appropriate. Hamel's recommendations were encoded by the facilitator into a 19 pattern PL. Since the group had no background in strategy analysis the PL was deliberately kept small and the initial patterns simple. Working on these patterns was expected to stimulate a wider discussion about the purpose and direction of CF.

The PL was worked on by the group over three days. Each pattern was introduced by the facilitator, here also acting as moderator. The context and forces as they affected CF were discussed. As in the earlier case it was necessary to translate in both directions, so that the group could understand what the pattern was saying, and then so that the pattern could subsequently be modified to be relevant to the needs of CF. This work was completed in two consecutive days. A week later the group gathered again to review their efforts. A number of changes were made. It was agreed that the one week gap had been helpful to the processes of reflection and revision. The facilitator's knowledge of business practice and theory and the novelty of these ideas for the group meant that the advisor had a much more influential and active role than was the case with TKC.

The language is shown in Figure 9 and a typical pattern in Figure 10.

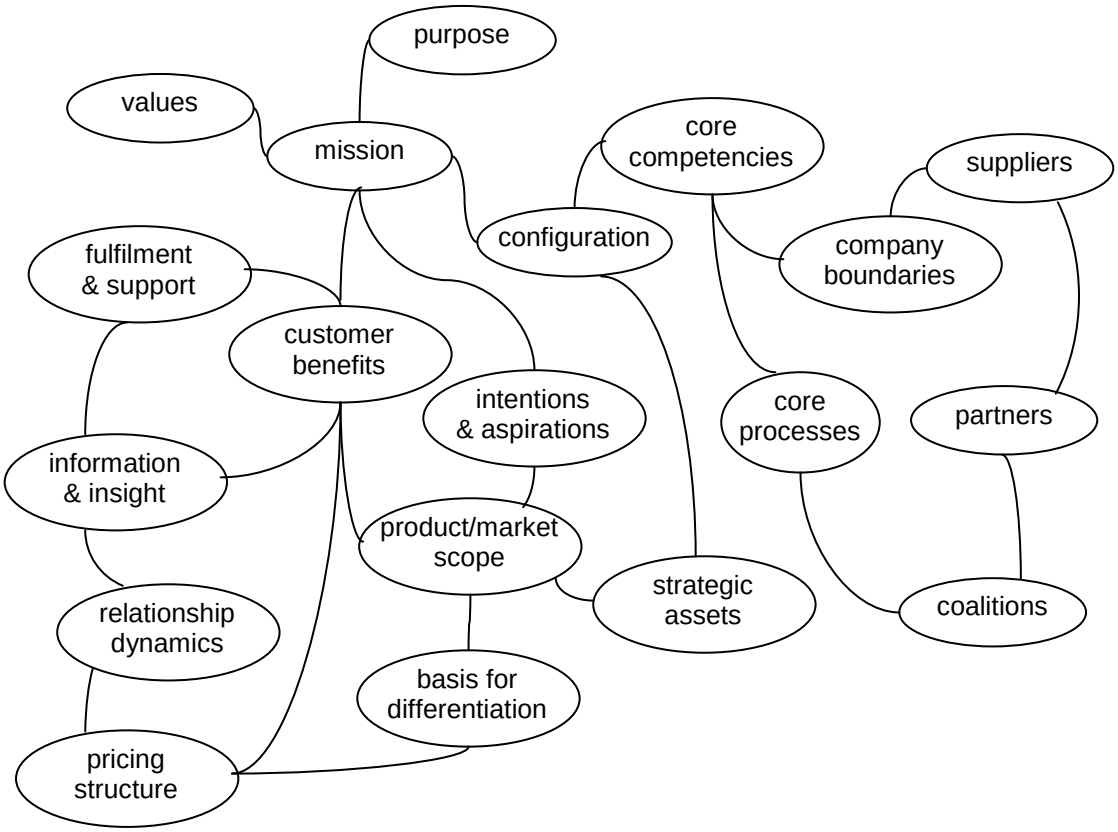

Figure 9. The CF PL map

Name:	**Product/Market Scope**
Aliases:	Who, What and Where.
Problem:	An organisation needs to clearly articulate its intended customers, geographic boundaries and product/service segments to be successful.
Context:	Customer demographics are continually changing, with geographic boundaries becoming less important as people and information flows more freely around the globe. Products/services tend to have a useful life cycle and need to be updated or replaced.
Forces:	Different customers want different things. A broader range of services costs more to supply. Market pressures conflict with interests of organisation.
Solution:	A document that captures the essence of where and to whom the organisation provides the "what" of service/product. The details should be arrived at after considering customer availability and needs.
Resulting Context:	An organisation that has clearly defined geographical areas of operation, clear customer target segments and a clear policy on what it will provide.
Rationale:	Limited resources mean that any organisation cannot produce every product or service demanded, clear limitations on scope must be set based upon the *Intentions & Aspirations.*
Links to other Patterns	***Intentions & Aspirations*** and ***Basis for Differentiation***

Figure 10. An example of a CF pattern.

The members had no background in strategy. Their concerns and experience at CF had been in the day-to-day running of the charity, looking only a short distance ahead while doing so. The need was quickly to acquaint them with some useful ideas about strategy, enough for them to feel confident in making decisions. In the initial two-day discussions the form of the patterns was key in achieving this. Discussing and modifying patterns forced them to consider context and forces – just why situations arise, what makes them problematic and why certain solutions might be useful. Considering a number of patterns quickly meant that this mode of thinking about problems soon became familiar to them. The same had been found at TKC.

Discussion

The cases described in this paper show how the use of donor patterns can speed learning in both an expert and a novice group. It is argued that the structure of patterns and the use of donor patterns was important in these comparatively short interventions. For TKC the donor patterns came directly from another business application and for CF they described some management theories (themselves based on observation and experience). Donor patterns were

used both as a way of foreshortening the interventions and also as ways of transferring knowledge.

The use of a PL, particularly the use of donor patterns, has implications for practice. Much of the literature on knowledge transfer and organisational memory is concerned with computer-based mediation. While such mediation works well in many circumstances (indeed, it may be the only option) our experience suggests that the social interactions of face-to-face group discussion using the structured approach of a PL is feasible and useful.

Use, re-use and learning

The PL literature is replete with examples of patterns proposed for use and much of the work of PL enthusiasts (particularly software developers) is concerned with the very careful process of pattern derivation and checking for validity and clarity among peer groups. Whether the resulting patterns are regularly used is not always clear. There is a little evidence: Zhang and Budgen (2012) found that only three of the 23 software patterns in the well known PL of Gamma et al. (1994) appear to be much used; Jacobson et al. (2002) recommend ten patterns for use in house design based upon the 250 proposed by Alexander et al. (1977), which admittedly had a much larger scope. This appears to be a characteristic of much knowledge management (Hult, 2003). This may be not simply because developing patterns is more fun than using them but because the learning gained during development is sufficient for the immediate purpose. While this is not at all what the proponents of PL had in mind it may be an accurate reflection of what people find useful.

The intention in a PL is to admit as patterns examples of repeatedly satisfactory solutions. Patterns have validity because they have been used several times. The confidence arising from this track record is meant to encourage reuse and reduce needless duplication (this was particularly a concern in the software community). However, successful implementation requires ownership. In the two cases described here it is the users directly who have learned and owned via comprehension and modification of the donor patterns. This may be enough. The idea of a repository of knowledge, while valid, may just be irrelevant for the user, as was the case with CF, because there is no intent at repetition. This is not to say that the CF experience could not itself form part of an augmented case collection.

In any case, repetition by users is usually not possible for managers. Programmers encounter problems frequently and so reuse is both possible and desirable. For managers this may not be so (how often is company strategy (re)designed?) which is why consultants provide a useful service as proxies for experience. Donor patterns provide this too.

Both the articulation and codification of knowledge are expensive in time and cognitive effort so that "in most cases articulated knowledge is never codified" due to the extra costs of

making manuals, or building decision support systems (Zollo and Winter, 2002). Pattern making, because of its modularity, has the potential to ease these difficulties.

Modular structures, here of a PL, can aid knowledge transfer and also innovation. For any given conceptual model individual modules (patterns in this case) may be altered in a number of ways: by *learning* from application; by *sharing* if procedures are adopted from other users; by *innovation* as the result of the adoption of a new procedure (Schilling, 2000; Schilling and Steensma, 2001). These are to some extent points on a continuum, but transferred (modified) practice from a different context can be innovative for the adopting organisation.

Size

The size of a PL is likely to be important in encouraging use. Harrison (1998) recommends that the number of patterns should be sufficiently small that they can be memorised because people are more likely to use them if they have them at their fingertips. He thinks about 25. Our PLs had 29 and 19 patterns. These sizes were not imposed by us; they arose as providing a satisfactory completeness for the group members.

Roles

In using donor patterns a number of roles have been identified:

moderator	introduce patterns and manage discussion about content, in particular manage the simultaneous development of conceptual and procedural models
advisor	help in the application of the derived patterns in the two cases this was limited, but more to the fore in CF clients may be their own advisors
expert	in both cases the person involved had expert knowledge – of chemical engineering for TCM and of business for CF
facilitator	a more general role encompassing both that of *moderator* and *expert*

Where the moderator is also an expert there must be a natural tendency to become an active participant, which could lead to bias.

A common view of facilitation is given by, for example, Mind Tools (2014). A facilitator should prepare well and then

design and plan the meeting

guide and control the process

ensure that outcomes are recorded and actioned

which corresponds to our definition. Schwarz (2002: Table 3.1) differentiates five types of facilitator. In the two cases our role was most similar to Schwarz's *Facilitative Consultant*,

someone who is not a group member but has expertise in process and content and who may be involved in decision making. We had some practice in this pattern process before the cases, so some expertise. We had no training as workshop facilitators though our various jobs (teacher, consultant) had provided experience of group facilitation in other environments.

The facilitation of dispersed virtual teams relying on computer assisted interactions using Group Support Systems (GSS) typically needs the skills of a facilitator familiar with this particular task. These people are likely to be expensive. An alternative is to use a collaboration engineer, who is responsible for designing the interaction which is then run by the group. A set of patterns, called thinkLets, has been designed to help in this and may be of use for running patterns workshops (Vreede et al., 2006; Kolfschoten et al., 2004, Horwitz and Santillan, 2012).

Process and timetable

The two groups of people and their organisational settings were quite different. The one common factor was time pressure in terms both of a tight deadline for the completion of the task and also the amount of time which could be devoted to the group sessions. In both cases the task took three days and was divided into three phases: discovery, review and presentation Table 1).

phase	description & purpose	duration (days)
discovery	learn about patterns understand donor patterns reject or modify	1½
review	consider modified patterns make further modification	1
presentation	think about how to present patterns to relevant audience	½

Table 1. Stages and durations

For TCM there were six half days each separated by a week. For CF there were two sessions separated by a week. The initial two-day session comprised discovery and an initial review. The final day completed the review and presentation.

To some extent these different timescales were dictated by the different pressures of the task. But they also are responses to the different characters of the groups. The professionals at TCM were able to use the frequent gaps between sessions usefully to reflect on the relevance

of the patterns to their practice. The group at CF had quickly to become familiar with the pattern process and so a more intense two-day session was appropriate.

It should be emphasised that the purpose of the two cases were different: at TKC a debriefing of experience while at CF a learning from the codified experience of others. The process was broadly the same for each: discussion of donor patterns to develop an understanding of the content and their applicability to the task at hand. At CF that was enough to form an informed base for a conventional discussion of strategy while at TKC modified patterns were produced.

Pitfalls

While using donor patterns makes short interventions feasible in time-pressured situations, using patterns derived by others leaves open the possibility of insufficient modification and consequent misapplication. Such a bias would be a manifestation of the anchoring and adjustment heuristic of Tversky and Kahneman (1974) in which the easily available starting point tends rarely to be modified to the extent that subsequent evidence – discussion in this case – would justify. The role of the advisor and moderator is key here.

In the work described the emphasis was on application rather than research. It would be useful to test for the presence of this bias in a controlled experiment.

Integration of partners

Being a member of a pattern-making group can provide an opportunity for people new to the task to become familiar with operational knowledge, as in the TCM case. There is also the potential to pool knowledge with business partners, notably those in the supply chain. In discussing Toyota, Dyer and Nobeoka (2000) note that suppliers participate in collective learning because it is better than trying to isolate their own proprietary knowledge. However, they also note that there is a risk of a drift to homogeneity in a knowledge sharing network so that new knowledge is less likely to be created. A convergence of view may indicate an agreed best practice, but the point still holds. It may be that the use of a PL debriefing can help to avoid thoughtless convergence.

Relation to other methodologies

There is a clear similarity between PL and CBR, PL being perhaps a form of soft CBR. The difference is the degree of abstraction required when using patterns. Comparing the two approaches Riss et al. (2007) note that PL needs more effort than CBR for two reasons; because of the abstraction inherent in patterns and because of the need to maintain and modify them based on experience (though they also point out that CBR cases must be reviewed for their continued relevance). Patterns place particular emphasis on the social aspects of

knowledge acquisition and use. While this makes their use more time consuming it also increases their value in providing a learning environment, as the two cases show.

The pattern approach also has similarities to Checkland's Soft Systems Methodology (SSM) (Checkland and Scholes, 1990). The four stages in SSM are

find out about the problem situation

formulate purposeful activity models

debate the situation

take action

This structure was used in the CF intervention. The Pattern Language was used in the second stage as a way of achieving group focus on the strategic problems facing the organisation. There was no intention to build a knowledge base for use in future activities by this group, only to provide an engagement with useful knowledge for the current task. It was unlikely that any of the participants would be in this situation again. In other design fields the domain experience which users bring to the task determines how they interact with the knowledge base and so their needs of it (Zhang and Budgen, 2012). In the two cases described above the role of the moderator differed in that for CF the knowledge, widely available, was codified as patterns by the moderator to facilitate the intervention while at TKC donor patterns were found in the literature.

Getting started

For those wishing to use the donor patterns approach four issues need to be addressed:

deciding the task

designing the workshops

finding donor patterns

deciding a pattern format

Tasks

It is unlikely that at any one time the whole cycle will be active. Three tasks are possible:

Task	Purpose	Outcome	Role
Debriefing	deciding what happened and codifying what worked	new or modified patterns	moderator
Learning	finding out about possible approaches to a new problem	understanding of problem and possible approaches	advisor
Deciding	using patterns to make a decision	solution method	facilitator

Table 2. Tasks and roles.

Decide who best fits the role. Perhaps someone external to the problem, or would familiarity be more useful? If full notes or a transcript are required this may be a second role.

The workshop

Aim for three days. How these are arranged is likely to depend on the availability of the participants. Consider their backgrounds: the more they have expertise and experience the more useful it may be to allow time for reflection between sessions. But practical considerations are likely to predominate. Be flexible.

In the cases reported here fairly small groups of three and four worked well and enabled quick progress. Larger groups may need more time and so are better avoided.

More conventional patterns workshops, not using donor patterns and writing patterns from scratch, are described by Coplien and Wolfe (2000) and Rising (1998: 83-84).

Finding donor patterns

This may prove to be difficult if the focus is not on process management.

For process patterns there are a number of models based on managing software development. These are likely to be good donors, as was the case with TKC, provided contextual differences are discussed. Coplien (1995) will be a good place to start. Harrison (1996), Weir (1998), and Taylor (1996) provide alternatives.

Most other patterns have been suggested for software design and developed in a series of annual meetings of PLoP (Pattern Languages of Programs) organised by the Hillside Group (Hillside Group, 2014a) and regional conferences. These activities have associated collections of patterns (Hillside Group, 2014b and 2014c; EuroPLoP, 2014) some of which may be of more general use (e.g. Laurier et al., 2009).

There do not seem to be patterns directly describing OR methods, though Herring (2002) uses a PL to articulate Stafford Beer's Viable System Model.

A good example of a pattern-like approach to quantitative modelling is the advice given to forecasters by Armstrong (2001). His object was to present useful evidence-based advice to those having to make forecasts. Evidence was obtained from published studies and collected in a large book, each chapter provided by experts in their field. The way in which the evidence was assembled is described in the book and by Armstrong and Pagell (2003). While the format follows that of a conventional text key advice is made plain as in this example from the chapter on role play:

■ **Describe their roles to subjects before they read the situation description.**

Roles affect subjects' perceptions of a situation. Babcock et al. (1995) had 47 pairs of subjects read their role instructions *before* reading the description of a law case and 47 pairs that read their roles afterward. The subsequent role-playing outcomes differed between these two groups.

Figure 11. Pattern-like advice (Armstrong, 2001)

While the book did not profess to be based on a Pattern Language one of the contributors was clear that it could be seen as part of the patterns movement (Collopy, 2003). The advice set continues to develop via the Forecasting Principles website.

It may be necessary to construct some donor patterns (as at CF). This will increase the preparation time though not the time needed for the group workshops. Although the pattern idea is to encode practices which have been successfully applied enough times to give confidence as to their usefulness it seems to us that useful advice may also be found from theory and from the stored experience of academics and consultants. These sources are not without difficulties, of confirmatory bias for instance, and so should be used carefully. In our usage the purpose of donor patterns is to provide a basis for critical discussion by adopters in the context of their own organisation and experience. They may reject the donor completely. It is part of the role of the facilitator to select only that which is believed to have merit.

Deciding a pattern format

Start with the Coplien model used here and discussed above in the *Pattern Language* section. Modify according to the needs of the situation. Not all patterns need have the same format; some may have more sections if implementation detail is included while those concerned with higher level topics may have fewer.

Consider whether the same form used for debriefing is best suited for communication. Perhaps the Alexandrian pattern may be better for your users.

Conclusion

Encouraging an active learning and transfer of knowledge to solve organisational problems is desirable. To do this the use of some form of modular structure is inevitable. This paper has presented the use of a Pattern Language and, in particular, of donor patterns as a way to achieve these goals.

These patterns enable learning through evaluation and modification rather than solely by a direct consideration of personal experience. While this process needs careful moderation it

does accommodate a style of learning which many people will recognise: critique and modification.

The speed of the process makes the use of patterns feasible in real world situations and the modular form permits organisational sharing.

References

Aamodt A and Plaza E (1994). Case-based reasoning: foundational issues, methodological variations, and system approaches. *A I Communications* **7**(1): 39–59.

Ackoff R L (1979). The future of OR is past. *Journal of the Operational Research Society* **30**(2): 93-103.

Ahn H J, Lee H J, Cho K and Park S J (2005). Utilizing knowledge context in virtual collaborative work. *Decision Support Systems* **39**(4): 563-582.

Alexander C (1964). *Notes on the Synthesis of Form*. Harvard University Press: Cambridge MA.

Alexander C (1966). A city is not a tree. *Design* **206** (February): 46–55.

Alexander C, Ishikawa S and Silverstein M (1977). *A Pattern Language*. Oxford University Press: New York.

Ambrosini V and Bowman C (2001). Tacit knowledge: some suggestions for operationalization. *Journal of Management Studies* **38**(6): 811–829.

Armstrong J S (ed) (2001). *Principles of Forecasting*. Kluwer Academic Publishers: Boston.

Armstrong J S and Pagell R (2003). The Ombudsman: Reaping benefits from management research: Lessons from the forecasting principles project. *Interfaces* **33**(6): 91-111.

Arkes H R, González-Vallejo C, Bonham A J, Kung Y-H and Bailey N (2010). Assessing the merits and faults of holistic and disaggregated judgements. *Journal of Behavioral Decision Making* **23**(3): 250-270.

Baldwin C Y and Clark K B (2000). *Design Rules: Volume 1. The power of modularity*. MIT Press: Cambridge MA.

Beer S (1985). *Diagnosing the System for Organizations*. Wiley: Chichester.

Benkler Y (2002). Coase's penguin, or, Linux and The Nature of the Firm. *Yale Law Journal* **112**(3): 369–446.

Brusoni S and Prencipe A (2006). Making design rules: a multidomain perspective. *Organization Science* **17**(2): 179–189.

Buschmann F and Meunier R (1995). A system of patterns. In: Coplien J O and Schmidt D C (eds). *Pattern Languages of Program Design*. Addison-Wesley: Reading, MA.

Buschmann F, Meunier R, Rohnert H, Sommerlad P and Stal M (eds) (1996). *A System of Patterns*. Wiley: Chichester.

Checkland P and Scholes J (1990). *Soft Systems Methodology in Action*. Wiley: Chichester.

Coad P and Yourdon E (1991). *Object-Oriented* Analysis. Prentice-Hall: Englewood Cliffs NJ.

Collopy F (2003). The Ombudsman: A pattern language for forecasters. *Interfaces* **33**(6): 89-90.

Cook S D N and Brown J S (1999). Bridging epistemologies: the generative dance between organizational knowledge and organizational knowing. *Organization Science* **10**(4): 381–400.

Coplien J O (1995). A generative development-process language. In: Coplien J O and Schmidt D C (eds). *Pattern Languages of Program Design*. Addison-Wesley: Reading MA.

Coplien J O and Harrison N B (2005). *Organizational Patterns of Agile Software Development*. Pearson Prentice Hall: Upper Saddle River NJ.
See also http://orgpatterns.wikispaces.com/ProjectManagementPatternLanguage

Coplien J O and Schmidt D C (eds) (1995). *Pattern Languages of Program Design*. Reading, MA.: Addison-Wesley.

Coplien J O and Wolfe B (2000). A pattern language for writers' workshops. In: Harrison N, Foote B and Rohnert H (eds). *Pattern Languages of Program Design 4*. Addison-Wesley: Reading MA.

Cunningham W (1996). EPISODES: A pattern language of competitive development. In: Vlissides J M, Coplien J O and Kerth N L (eds). *Pattern Languages of Program Design 2*. Addison-Wesley: Reading MA.

Darr E D and Kurtzberg T R (2000). An investigation of partner similarity dimensions on knowledge transfer. *Organizational Behavior and Human Decision Processes* **82**(1): 28-44.

Dearden A and Finlay J (2006). Pattern languages in HCI: A critical review. *Human-Computer Interaction* **21**(1): 49-102.

Dyer J H and Nobeoka K (2000). Creating and managing a high-performance knowledge-sharing network: the Toyota case. *Strategic Management Journal* **21**(2): 345–367.

Easterby-Smith M, Lyles M A and Tsang E W K (2008). Inter-organizational knowledge transfer: current themes and future prospects. *Journal of Management Studies* **45**(4): 677-679.

Edmonson A C, Winslow A B, Bohmer R M J and Pisano G P (2003). Learning how and learning what: effects of tacit and codified knowledge on performance improvement following technology adoption. *Decision Sciences* **34**(2): 197–223.

EuroPLoP (2014). Europlop patterns repository. http://www.europlop.net/content/repositories, accessed 2 July 2014.

Fowler M (1997). *Analysis Patterns: Reusable Object Models*. Addison-Wesley: Reading MA.

Gabriel R P (1996). *Patterns of Software: tales from the software community*. Oxford University Press: New York.

Gamma E, Helm R, Johnson R and Vlissides J (1994). *Design Patterns: Elements of Reusable Object-Oriented Software.* Addison-Wesley: Reading MA.

Hall M (2006). Knowledge management and the limits of knowledge codification. *Journal of Knowledge Management* **10**(3): 117-126.

Hamel G (2000). *Leading the Revolution*. Harvard Business School Press: Boston MA.

Harrison N B (1996). Organizational patterns for teams. In: Vlissides J M, Coplien J O and Kerth N L (eds). *Pattern Languages of Program Design 2*. Addison-Wesley: Reading MA.

Harrison N B (1998). Potential pitfalls, or how to jump on the patterns bandwagon without the wheels coming off. In: Rising L (ed.) *The Patterns Handbook: techniques, strategies and applications*. Cambridge University Press: Cambridge.

Harrison N, Foote B and Rohnert H (eds) (2000). *Pattern Languages of Program Design 4*. Addison-Wesley: Reading MA.

Hatch M J (1999). Exploring the empty spaces of organizing: how improvisational jazz helps redescribe organizational structure. *Organization Studies* **20**(1): 75–100.

Herring C E (2002). *Viable Software: The intelligent control paradigm for adaptable and adaptive architecture* . PhD Thesis, The University of Queensland . (Ch 4). http://charlesherring.com/thesis/ViableSoftwareContentsandChapter1.pdf **,** accessed 8 July 2014.

Hillside Group (2014a). PloP14. http://www.hillside.net/plop/2014/ , accessed 2 July 2014.

Hillside Group (2014b). Patterns Library. http://hillside.net/patterns/177-patterns-library , accessed 2 July 2014.

Hillside Group (2014c). Patterns Catalog. http://hillside.net/patterns/patterns-catalog , accessed 2 July 2014.

Hoetker G (2006). Do modular products lead to modular organisations? *Strategic Management Journal* **27**(5): 501–518.

Horwitz S K and Santillan C (2012). Knowledge sharing in global virtual team collaboration: applications of CE and thinkLets. *Knowledge Management Research and Practice* **10**(4): 342-353.

Hult G T M (2003). An integration of thoughts on knowledge management. *Decision Sciences* **34**(2): 189–195.

Jackson P (2012). Transactive directories of organizational memory: towards a working data model. *Information & Management* **49**(2): 118-125.

Jacobson M, Silverstein M and Winslow B (2002). *Patterns of Home: The Ten Essentials of Enduring Design*. The Taunton Press: Newtown CT.

Kamoche K, Pina e Cunha M and Vieira da Cunha J (2003). Towards a theory of organizational improvisation: looking beyond the jazz metaphor. *Journal of Management Studies* **40**(8): 2023–2051.

Kingston J and Macintosh A (2000). Knowledge management through multi-perspective modelling: representing and distributing organizational memory. *Knowledge-Based Systems* **13**(2-3): 121-131.

Kolb D A and Fry R (1975). Toward an applied theory of experiential learning. In: Cooper C (ed). *Theories of Group Process*. Wiley: London.

Kolfschoten G L, Briggs R O, Appelman J H and Vreede G-J de (2004). Thinklets as building blocks for collaboration processes: a further conceptualisation. In: Favela J and Decouchant D (eds). *Groupware: Design, Implementation and Use*. Springer: Berlin.

Kolodner, J (1993). *Case-Based Reasoning*. Morgan Kaufmann: San Mateo CA.

Laurier W, Hruby P and Poels G (2009). Business plan conception pattern language. *EuroPLoP 2009.* https://biblio.ugent.be/input/download?func=downloadFile&recordOId=910647&fileOId=910663 , accessed 8 July 2014.

Manns M L and Rising L (2005). *Fearless Change: Patterns for Introducing New Ideas*. Addison-Wesley: Boston, MA.

Meszaros G (2014). *A Pattern Language for Pattern Writing*. http://hillside.net/index.php/a-pattern-language-for-pattern-writing , accessed 14 July 2014.

Meszaros G and Doble J (1998). A pattern language for pattern writing. In: Martin R, Riehle D and Buschmann F (eds). *Pattern Languages of Program Design 3*. Addison-Wesley: Reading MA.

Miller G A (1956). The magical number seven, plus or minus two: some limits on our capacity for processing information. *Psychological Review* **63**(2): 81–97.

Mind Tools (2014). *The Role of a Facilitator*. http://www.mindtools.com/pages/article/RoleofAFacilitator.htm , accessed 4 December 2014.

Nelson R and Winter S (1982). *An Evolutionary Theory of Economic Change*. Harvard University Press: Cambridge MA.

Nevo D and Wand Y (2005). Organizational memory information systems: a transactive memory approach. *Decision Support Systems* **39**(4): 549-562.

Nonaka I (1994). A dynamic theory of organizational knowledge creation. *Organization Science* **5**(1): 14–37.

Orlikowski W J (2002). Knowing in practice: enacting a collective capability in distributed organizing. *Organization Science* **13**(3): 249–273.

Papalambros P Y and Wilde D (2000). *Principles of Optimal Design: Modelling and Computation*. Cambridge University Press: Cambridge.

Pedagogical Patterns Editorial Board (2012). *Pedagogical Patterns: Advice for Educators*. Joseph Bergin Software Tools.

Rips L J (1989). Similarity, typicality and categorization. In: Vosniadou S and Ortony A (eds). *Similarity and Analogical Reasoning*. Cambridge University Press: Cambridge.

Rising L (ed) (1998). *The Patterns Handbook: techniques, strategies and applications*. Cambridge University Press: Cambridge.

Riss U V, Cress U, Kimmerle J and Martin S (2007). Knowledge transfer by sharing task templates: two approaches and their psychological requirements. *Knowledge Management Research and Practice* **5**(4): 287–296.

Rogers D and Salustri F A (2009). A quality function deployment method pattern language for efficient design. In: Bergendahl N, Grimheden M, Leifer L, Skogstad P and Lindemann U (eds). *Proceedings of ICED 09, the 17th International Conference on Engineering Design, Vol. 5, Design Methods and Tools (pt. 1)*. The Design Society, Palo Alto CA.

Salingaros N A (2000). The structure of pattern languages. *Architectural Research Quarterly* **4**(2): 149–162.

Sanchez R and Mahoney J T (1996). Modularity, flexibility, and knowledge management in product and organisation design. *Strategic Management Journal* **17**(Special Issue: Knowledge and the Firm, Winter): 63–76.

Schank R (1982). *Dynamic Memory; a theory of reminding and learning in computers and people*. Cambridge University Press: Cambridge.

Schilling M A (2000). Towards a general modular systems theory and its application to interfirm product modularity. *Academy of Management Review* **25**(2): 312−334.

Schilling M A and Steensma H K (2001). The use of modular organizational forms: an industry-level analysis. *Academy of Management Journal* **44**(6): 1149−1168.

Schindler M and Eppler M J (2003). Harvesting project knowledge: a review of project learning methods and success factors. *International Journal of Project Management* **21**(3): 219-228.

Schwarz R (2002). *The Skilled Facilitator: A Comprehensive Resource for Consultants, Facilitators, Managers, Trainers, and Coaches*. Jossey-Bass: San Francisco.

Slywotzky A J, Morrison D J, Moser T, Mundy K A and Quella J A (1999). *Profit Patterns: 30 Ways to Anticipate and Profit from Strategic Forces Reshaping Your Business*. Wiley: Chichester.

Stein E W and Zwass V (1995). Actualizing organizational memory with information systems. *Information Systems Research* **6**(2): 85-117.

Szulanski G (2000). The process of knowledge transfer: a diachronic analysis of stickiness. *Organizational Behavior and Human Decision Processes* **82**(1): 9-27.

Taylor P (1996). Capable, productive, and satisfied: some organizational patterns for protecting productive people. In: Vlissides J M, Coplien J O and Kerth N L (eds). *Pattern Languages of Program Design 2*. Addison-Wesley: Reading MA.

Tversky A and Kahneman D (1974). Judgement under uncertainty: heuristics and biases. *Science* **185**(4157): 1124–1131.

von Hippel E (1994). "Sticky information" and the locus of problem solving: implications and innovation. *Management Science* **40**(4): 429–439.

Voss C A and Hsuan J (2009). Service architecture and modularity. *Decision Sciences* **40**(3): 541–569.

Vreede G-J de, Kolfschoten G L and Briggs R O (2006). ThinkLets: a collaboration engineering pattern language. *International Journal of Computer Applications in Technology* **25**(2/3): 140-154.

Weick K E (1998). Improvisation as a mindset for organizational analysis. *Organization Science* **9**(5): 543–555.

Weir C (1998) Patterns for designing in teams. In: Martin R, Riehle D and Buschmann F (eds). *Pattern Languages of Program Design 3*. Addison-Wesley: Reading MA.

Williams C (2007). Transfer in context: replication and adaptation in knowledge transfer relationships. *Strategic Management Journal* **28**(9): 867-889.

Zander U and Kogut B (1995). Knowledge and the speed of the transfer and limitation of organizational capabilities: an empirical test. *Organization Science* **6**(1): 76-92.

Zhang C and Budgen D (2012). What do we know about the effectiveness of software design patterns. *IEEE Transactions on Software Engineering* **38**(5): 1213-1231.

Zollo M and Winter S G (2002). Deliberate learning and the evolution of dynamic capabilities. *Organization Science* **13**(3): 339–351.

Appendix

The following two fragments of TKC workshop transcripts illustrate the use of donor patterns M is the moderator and A - D, are the group members. {…} indicates discussion.

Illustration 1. A donor pattern is being considered. The donor was from a software development environment and it was unclear how relevant it was to TKC. A new pattern is generated as a reaction to the donor.

M: What is the analogue of the testing issue in your context?

A: It's very important for us to understand what we need to test or pilot. We need clarity on whether and at what level, and then plan it in. {implications}

M: So is this relevant.

B: It's not relevant in this form, but there is a process to determine what is to test.

C: The problem applies (not the context), but the solution for us is totally different.

B: For us it is governed by the internal process, which demands you do certain things before you commit. {detail}

M: That answers the question about when to test?

C: Highlight risks with customer, operational risk, other techniques.

then after more discussion …

B: I thought this pattern was about deciding how we, as an R&D group, decide what we need to test; and I haven't heard a solution yet because you two [A and C], know because you've done it before.

A: We had an approach, C will lift it out and package it; an action.

B: And then there's this other one about having decided that, we have to go out to a customer…

D: Yes but we've already discussed a pattern for engaging customers, which must be related to this, and so we don't have to discuss it in the context of this pattern.

A: For Project 1, we had all the customers and operations people in house. In the new world, who is the operations team? Who will own and run the test on a plant?

{lengthy debate over how to get credibility to test early technology, bringing in customers etc. again}

D: … there's no great problem with this. We've discussed at least one other pattern to address it … Engage customers – it's all there: "closely couple the customer to the developer and architect, not just QA", to me that made sense. This pattern is specifically about <u>testing</u>.

{more debate about teams, compositions, roles etc.}

A: {Summary and clarification statement; comparison of old context, and new context}

M: And they need role clarity on their side…So now you've managed to define some guidelines for a solution to this – that's your new pattern.

Illustration 2. The discussion here is about team selection. The donor pattern provokes a discussion at a fairly detailed (procedural) level and this leads to the generation of a new pattern (conceptual level).

M: Looks like a lot of meat in there – a variety of solutions based on context. This "self-selecting" name pre-supposes a solution for you.

A: It may be stating the obvious, but the team has to get on – it has to be a team. If it doesn't have a team spirit, it's not going to happen as project.

B: What are we saying here? There's a problem selecting the team, but self-selection isn't the answer, and this other solution is? Is that what you did on Project 1?

A: It was a mixture. We started with self-selection, people got on and congregated into clusters, but there was some specific recruitment for competence. And the psychometrics came in when the team was mostly complete, and we used it to help internal processes. It was a little late in a way, but the team suddenly understood why certain things were happening.

C: I'd agree with that. I think using Kurtin index was good practice, not necessarily for team selection, but to help people understand themselves and their role, and what skills they bring. You can rationalise the conflicts. I think that's good practice.

A: That's a slightly different related point.

B: Haven't you generated yourself another pattern there? One is "Select the team", and then "How do you get them to work together". I don't know if there is a pattern for that.

M: There is a pattern called "Team Spirit", so just add one in that vein.

A: Yes "Team Building" [awaydays, courses etc.] were an important part of the project.

C: We certainly need to build functional engineering in later, so we need an approach to influencing that.

B: So now we're talking about another pattern, which is "How do you manage changing resource needs".