Describing a design pattern: Why is it not enough to identify patterns in educational design?

Christian Voigt, Paula M.C. Swatman
School of Computer and Information Science
University of South Australia

In this paper we analyse the use of patterns across a number of fields including architecture, software development and educational technology design. Focusing on the reusability of a pattern outside its area of development, we have identified several issues related to the context and the value system of a pattern. The paper draws together lessons learned from different fields where patterns are already used and described. We conclude with a recommendation of pattern descriptors and guidelines which improve their applicability in varying value systems.

Keywords: design patterns, pattern format, design complexity, research and design values

Introduction

The term 'pattern' as used in this article was first defined by Alexander (1979) to mean a recurring problem-solution pair which can be observed under a variety of conditions. Alexandrian patterns in their most essential form consist of problem, solution and context descriptions; and this approach to patterns has been widely accepted by design sciences such as architecture, software development (Gamma, Helm, Johnson & Vlissides, 1995) and educational technology design (Derntl & Motschnig-Pitrik, 2005; Goodyear, 2005; Niegemann, Hessel & Domagk, 2004). Patterns, however, also have the potential to assist online educators in a broader sense during their daily activities – an important goal, because although online education design models are being created in increasing numbers, they are meeting with increasingly less acceptance from design practitioners (Conole, Dyke, Oliver & Seale, 2004). Niegemann et al. (2004), for example, suggest that design patterns have the potential to represent theory-informed design in a way that is closer to the everyday experience of designers and instructors looking for solutions to their specific problems, rather than purely as all-embracing, abstract models – which is how they are generally viewed at present.

Research objective

This paper seeks to analyse the way patterns are described and to show how such description affects their reusability in other, later designs. We argue that there is a need for greater transparency about what needs to be described (and why). We believe that pattern descriptions include a number of assumptions which, if not made explicit, limit their future dissemination, application and evaluation.

The paper begins with a brief discussion of pattern applications in architecture, software development and educational technology, including examples and domain-specific issues. To analyse the identified issues constructively, we next consider patterns as products of cultural as well as academic values; and provide examples of why it is important to differentiate these two types of values. We then discuss the process of applying patterns in a context-sensitive way and why patterns offer a potentially helpful approach to coping with design complexity. Finally, we match the issues identified against existing pattern descriptors and suggest a more comprehensive format for pattern description.

Pattern definitions and usage

In this section we review the current pattern-related discussions in the disciplines of architecture, software development and educational technology. Taking into account the fact that patterns have been discussed for more than 25 years in architecture and for over a decade in software development (Gamma, Helm, Johnson & Vlissides, 1995), however, we have had to limit this review to the works of a comparatively small number of authors – those who seem to encapsulate the views of pattern researchers within their field and who have been most frequently cited by later authors.

Architecture

Architectural patterns were first promoted by Alexander and colleagues in the late 1970s through their book '*A timeless way of building*' (Alexander, 1979):

> Every pattern we define must be formulated in the form of a rule which establishes a relationship between a context, a system of forces which arises in that context, and a configuration which allows those forces to resolve themselves in that context (Alexander, 1979, p.252).

In this definition, Alexander identifies the principal relationships of his pattern approach in terms of solution, problem and context.

Solution
A pattern describes a solution to an archetypal problem in a general way, but is not a recipe which can be applied without understanding the relationships described within the pattern itself; and between the pattern and its neighbouring patterns (Alexander, 1979, p.223). Patterns still need to be adapted to local conditions – they are a kind of modular knowledge which refers to other patterns at different levels of abstraction, best represented in terms of scalar relationships (Salingaros, 2000). Alexander takes an inductive approach to pattern mining: starting from observations and the idea of a good quality design, he abstracts the essential features of that design. These features will be the core elements of the suggested solution.

Problem
After identifying what captures the quality of a design (solution) in a particular situation, Alexander describes the problem in terms of forces dominating the scene (problem description). Forces are constraints on the quality of a pattern.

Context
Alexandrian patterns require contextual specifications for when the problem is most likely to be encountered and when the suggested solution is most likely to be successful.

Software development

Though the link between Alexandrian architecture and software engineering goes back as far as 1968 (Coplien & Devos, 2000), wide-spread adoption of design patterns only occurred when object oriented programming became the dominant approach (Gamma, Helm, Johnson & Vlissides, 1995).

Not all patterns, however, need to be object oriented. Coplien (1997), for example, takes a broader approach and notes that the definition of a pattern is (again in the Alexandrian tradition) the solution to a problem in a context, whatever that solution might be – reaching beyond classes, objects and methods:

> Software pattern practitioners have turned to patterns as reaction against this unsatisfying aspect [the formal and scientific grounding of software design with no reference to any value system] ... (Coplien, 2004, p.7).

Coplien's and Gamma et al.'s statements about patterns highlight some issues accompanying the usage of software design patterns: 'What counts as a pattern?' and 'What contributions to the design process can be expected from patterns?'

A much more open approach to the use of patterns is taken by Fowler (1997), who sees them as a way of formulating suggestions which may or may not be accepted by his clients. Regardless of whether a pattern is accepted or not, he emphasises, patterns are about clients getting to know their own systems. He argues that patterns have only a limited potential to be applied outside the context in which they were created and he delegates final decisions to the respective domain experts. Fowler also points out that the validity of a pattern depends on the use of a common framework, which defines the conditions under which it can be used in the future. If we take the reuse of software components as an example, such a framework could be a Visual Basic environment, or object oriented technology. As we show in the next section, Goodyear (2005) similarly argues for a pedagogic framework as strategic/ tactical guidance for applying patterns in e-learning.

Educational technology

One reason for introducing different pattern forms in educational technology design is to distinguish between the representational media used to implement them: *computer languages* (highly formalised representations, such as mark-up languages or programming languages), *meta-languages* (e.g. UML, E²ML); and *natural language* descriptions (such as narratives or tables). This can be seen as an attempt to represent patterns in a human as well as a machine-readable format.

Computer languages
Efforts to create libraries of templates and learning resources link design patterns to the reuse of learning objects in different contexts, suggesting either a best practice *sequence* of learning objects, or a best practice *combination* of learning resources (Green, Jones, Pearson & Gkatzidou, 2006a). Adaptation to context is mainly seen as matching learning object meta-data with learner profiles (Green et al., 2006a). Though patterns can be transformed into physical representations, it is claimed that the use of highly automated patterns may: (a) impede adaptation, since this would imply additional encoding effort; (b) constrain the creativity of designer who generally prefers to take 'what's available' than 'what's most appropriate'; or (c) lead to a mismatch between the learning situation and pattern due to an overly general specification of pattern-related conditions and restrictions (Green, Jones, Pearson & Gkatzidou, 2006b; Heath, 2006). The last point, in particular, highlights the fundamental question of the degree to which a learning situation can be analysed algorithmically (Dreyfus, 1992).

Meta-languages
Motschnig-Pitrik & Derntl (2005) stress the importance of meta-languages such as UML to mediate between pedagogic design and software design. Care is needed, however, when using UML as the specification tool of choice because, although UML models reduce complexity by means of abstraction and modelling perspectives such as use case views, activity views or interaction views (Rumbaugh, Booch & Jacobson, 1999), parts of the semantic information included in the UML diagrams may remain hidden if the reader is not familiar with UML syntax. Modelling is a means to an end and there is a clear need to match pattern representations with their target audiences. UML was developed to support object-oriented software development (Rumbaugh, Booch & Jacobson, 1999) and the benefits of using UML to support design and diffusion of educational technology patterns outside IT-related education remain to be seen.

Natural language
Peter Goodyear (2005) identifies design patterns as new conceptual tools requiring a demand analysis. Demanded are "customizable, re-usable ideas ... there is no visible demand for complex methodologies …" (Goodyear, 2005, p.82). This approach is different from the two previous ones because Goodyear begins by sharing design ideas whose primary representation is textual. The benefit of a textual representation is that it is unlikely to impose itself on the designer, since alterations are made easily. Textual representations also have a wider potential audience, which at some stage may even include the learners themselves.

Cross domain issues and remedies

As the previous section has shown, the concept of design patterns has been embraced by designers from a number of different backgrounds. No matter whether patterns as a design construct were widely accepted and adopted by a research community as has been the case with software developers (Gamma, Helm, Johnson & Vlissides, 1995), or were viewed with less enthusiasm – as was the case with architects (Salingaros, 2000), issues relating to their application reappear in surprisingly similar forms across the various domains. Based on our review of pattern literature, we identified two sets of issues:

- the definition of patterns which usually originate in an environment with a homogeneous value system, including *institutional, pedagogic* and *methodological* values;
- the application of patterns in complex design environments; and
- the way patterns can adapt to emergent changes in context.

Institutional values: Identify cultural idiosyncrasies

The way patterns are understood and used has a significant impact on the way they are evaluated. Wottawa & Thierau (1998) point out that evaluations generally support decision-making processes by placing a value on alternatives. Commonly-held quality standards of the stake-holders involved therefore have a significant influence on the evaluation process.

Schein (1996) defines culture in institutions as a combination of visible norms, exposed values and deeply held assumptions. Applying Schein's concept of culture to patterns as design artefacts, two questions emerge: (a) do the inherent values of the patterns align with the users' culture(s)? and (b) what do we know about the users' culture(s) in the first place?

For example, Anderson, Plessis & Nickel (2001) describe a distance collaboration project by educators from a South African and a North American University: after the initial contact it was intended that a more in-depth discussion via asynchronous forum would take place. The authors reported, however, that this discussion process did not succeed because faculty members from the two universities valued online discussions very differently: while the North Americans saw online forums as a low-effort mode of communication, the South African institution – with little experience and an unreliable technical infrastructure – found it much more taxing.

Axiological assumptions: Make the pedagogic pattern value base explicit

Axiology is the theory of values (i.e. a group's moral, ethical and aesthetic assumptions) and determines what drives the design process (Banathy & Jenlink, 2003). Alexander was striving for universally valid patterns and Grabow (1983) supported Alexander's attempt to achieve a Copernican revolution in architecture. Though Alexander's assertion that a "purely intuitive approach to architectural design was no longer capable of adequately responding to the complexities of industrialization, urbanization and social change" (Grabow, 1983, p.6) may be viewed with sympathy, it is not clear that this issue can be resolved through the application of a single theory. Protzen (1978) points out that many statements in Alexander's pattern description are either inconsistent or of little empirical value for the formation of a coherent theoretical system; and then invokes the support of Feyerabend (1988), who criticises 'all-encompassing-theories' which, by excluding alternative views, close themselves to any criticism and run the risk of becoming ideologies.

In the case of educational technology patterns, values come into play when the instructional designer defines the more abstract conditions of the educational environment. Goodyear (2005) combines the more abstract elements of design into a pedagogical framework which represents the conditions and intentions for: the formulation of actual learning tasks; the definition of system features; and the organisation of group work. Though some patterns may be independent of any pedagogical background, others change their meaning if interpreted within a different educational paradigm (Marshall, 1996). A good example is assessment-related patterns, where the instructors' understanding of learning, valuable learning objectives and the learners' role directly influence their assessment pattern. Though all instructors may be using collaborative, problem-based learning, those working within a social constructivist value system will place greater emphasis on the student's own contribution to the learning process. By contrast, instructors using cognitive approaches will believe that the assessment focus should be on the level of acquired knowledge as the most reliable source of evidence for whether learning has taken place (Savin-Baden, 2004). An explicit description of the underlying pedagogic values of a pattern may also strengthen the position of an instructor who is asked to explain or justify a specific learning activity in combination with a given communication channel to get students fully engaged.

Research paradigms: Include data and methods applied

In the previous two sections, we have pointed out the potential which cultural, institutional and pedagogic values may have to prevent the success of a pattern – if not made explicit. The last set of homogeneous values critical to the definition of patterns is those inherent in the various research paradigms. Research paradigms are described by Habermas (1971) as 'Erkenntnisinteresse' (the quest for knowledge and cognition) and this is also true for the formulation of patterns. Before we can describe a pattern, we need to clarify its intention, to set the scene for its interpretation. A hypothetical pattern based on research results from computer mediated communication could follow one (or more) of the following three perspectives based on a positivist, interpretivist or critical research paradigm:

Positivist view
According to Popper (2002), scientific theories are about universal (time and location independent) statements which can be tested inter-subjectively: the higher the empirical content of the theory, the more precise that theory (Popper, 2002/1935, pp.4–22). Researching the use of communication technology from a positivist perspective would thus involve, in the most extreme form of this view: "quantifiable independent and dependent variables, and hypothesis testing, typically involving laboratory experiments and statistical inference" (Ngwenyama & Lee, 1997, p.149). A less extreme view of positivism would suggest an externally discoverable truth resulting from any research undertaken within the positivist tradition. The positivist perspective has a clear view of what is considered objective data.

Interpretivist view
Interpretivist studies take the approach that all truth is relative – interpretivist researchers do not believe that there is any externally discoverable, absolute truth. Interpretivists believe that the world is interpreted via the mind of the observer and that the language and symbols used by those describing what they perceive are an inherent part of that observed reality (Weber, 1922). Clearly, then, the concept of 'objective' data has little meaning for interpretivist researchers since they view all information as subjective.

Critical theory
Taking a Marxist perspective of research, critical theory opposes the definition of knowledge as 'knowledge of control' and argues for the inclusion of 'reflective knowledge' (Habermas, 1971, p.47), believing that while the former is crucial for technical processes, the latter dominates social processes. Critical theory requires researchers to question the social and historical conditions of what they observe and to uncover restrictions which may originate from accepting the status-quo (Orlikowski & Baroudi, 1991). Consequently, it is not sufficient to measure or explain circumstances – researchers must also 'emancipate' from them (Ngwenyama & Lee, 1997).

These three perspectives can perhaps best be compared by means of an example. A scripted collaboration pattern may have: a *positivist version*, focusing on the impact of what is technically possible to enhance collaboration; an *interpretivist version*, focusing on the group context to monitor and understand collaboration; or a *critical version,* focusing on changing the context – because a reward system which is geared towards measuring individual performance does not encourage collaboration in the first place.

The inclusion of this paradigmatic dimension has the advantage of clarifying why a pattern relies on certain data and how those data were obtained: and could greatly improve the credibility of educational technology design patterns.

Context: Provide positive and negative cases

Applying a design pattern to a real world problem means matching what is described as a workable solution to one's own specific problem. Since, in a socio-technical system such as an online learning environment, it is unlikely the same problem will occur in exactly the same context as that in which the pattern was developed, designers who would like to (re-)use patterns are confronted with implementation issues such as the contextual dependence of a pattern and the complexity characterising the design problem.

Context plays an important role in determining the adequacy of a pattern: architectural patterns, for example, are dependent on climatic conditions (Protzen, 1978) or institutional norms and rules (Bryant, 1994); and software patterns may depend on programming paradigms (Schmidt, 1995). So how do these differences in context affect the utility of a pattern? When should a pattern be discarded, or its contextual features changed? Context is also the issue which ensures the relationship of a pattern to an existing problem, e.g. how to ensure meaningful collaboration to a tested solution among all members of a group (Voigt & Swatman, 2006); or scripting a collaboration process so as to leave enough room for creativity and thus avoid scripted activities simply being ticked off and stripped of their educational purpose (Dillenbourg, 2002).

In deciding on the relevance of a particular context feature, not only should the optimal conditions for a pattern be stated, but also conditions under which the pattern has failed, or might fail. Consequently, to continue the example mentioned above, a script-using pattern might include information about when not to apply the pattern, e.g. when a learning community extends over different time zones, or uses highly heterogeneous technologies. Preserving critical cases which did not conclude as predicted are common practice in multiple case studies, where a theoretical framework is strengthened by theoretical replication (Yin, 2003, p.47). Similarly, a pattern may include critical circumstances under which the pattern failed, showing the need to either create the 'necessary' conditions, or to change the pattern itself.

Complexity: Provide concrete examples and abstract framework

The more contextual conditions and forces are contained in a pattern description, the more complex it becomes to apply the pattern to a concrete design problem. In this section we look at educational scenarios as complex systems and attempt to show how patterns provide a suitable approach for interacting with complex design problems. Emmeche (1997) differentiates between descriptive and ontological complexity: the former describes situations where various models and methods are needed to give a reasonably complete account of the phenomena under research; while the latter refers to situations in which, although there are discernable rules within the system, these are not predictable. Non-predictable, complex systems are typically characterised by 'emergences', i.e. not all higher level phenomena can be deduced from lower level phenomena, thus creating new patterns in a system's behaviour (Goldstein, 2004).

Ontological or emergent complexity poses a problem for the prescriptive power of design patterns. Though they capture 'proven' design knowledge, new designs need to be monitored so that the researcher may become aware of emergent, unforeseen side effects. Emergent complexity may be dealt with by giving extensive examples which illustrate differing instantiations of a pattern. Concrete examples support the understanding of patterns as partial views of complex situations which adopt different perspectives. Examples have the benefit of linking together otherwise disjoint, abstract components of formal descriptor categories in a narrative way.

To obtain the full benefit from these narratives, designers need to enter into a hermeneutic dialogue with a pattern and its examples. A fundamental principle of hermeneutics as represented by Gadamer (1994) is the '*hermeneutic circle*' of interpretation: "we must understand the whole in terms of the detail and the detail in terms of the whole" (p.291). Applying this approach to the interpretation of patterns: 'we understand the pattern through examples and the example through the pattern'. A major benefit of design patterns is that they allow for the *reconstruction* of complexity and context by the designer due to the mix of abstract models, real examples; and hermeneutic dialogue.

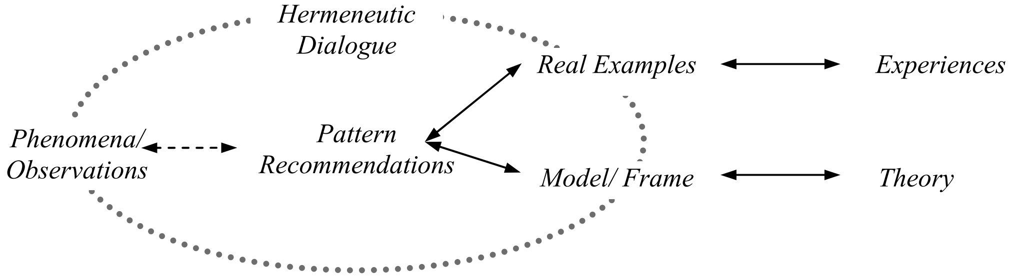

Figure 1: Pluralistic pattern interpretations

The dashed line in Figure 1 indicates the incomplete relationship between the enactment of a design situation and the predictions of a pattern – 'incomplete' because of complexity and contextual issues as discussed in the two previous sections. By claiming less generalisability and providing more contextual information, patterns actually enable designers to make some design decisions on their own. A pattern's value does not lie in its suggestions alone, but also in its support of a hermeneutic dialogue and the designer's learning to use the pattern in a context-sensitive way.

Suggested pattern descriptors

In previous sections we identified a need to understand, refine and possibly adapt a pattern when applied in a different context. In order to provide the user of a pattern with the necessary means to do so, we have expanded the traditional triplet of *context*, *problem* and *solution* with a fourth category *evaluation*. We are aware that this represents a change in the use of the pattern format which so far has been understood to capture 'approved' knowledge. However, since the dissemination of patterns in areas others than software development has proven to be problematic; we argue, that users need the means to evaluate the performance of a pattern by themselves. Table 1 is a synthesis of the various pattern description formats used in architecture, software development and educational technology design; and provides a useful tool for would-be re-users of patterns – as well as developers of new patterns in these areas.

Table 1: Pattern descriptors

Category/ descriptor	#	Explanatory question
Context		
Problem context	1	Under which circumstances the problem is likely to occur?
Pattern conditions	2	Under what conditions the suggested solution should or should not be applied?
Educational paradigm	3	What is the epistemological frame of reference to formulate a Pattern?
Domain specifics	4	What contextual characteristics need to be preserved (usually the invariables)?
Composite patterns	5	How does the pattern complete a larger pattern?
Networked patterns	6	Any other patterns making use of or being used by the pattern.
Similar patterns	7	Which patterns are closely related and how are they different?
Problem		
General description	8	What describes the problem?
Conflicting forces	9	Which are the forces in conflict?
Problem example	10	What are examples of poor design?
Problem scope	11	At what scale does the conflict occur?
Deepness	12	How deep does the conflict reach?
Substance	13	What are the structures and activities involved (physical environment, activities)?
Solution		
Name / Title	14	What summarizes the solution?
General description	15	What describes the solution?
Introduction	16	What is the pattern's intent?
Aliases	17	What are other well-known names for the pattern?
Principle	18	What general principles give base to the solution?
Pattern visualization	19	What displays the pattern best?
Structural Changes	20	What characterizes the physical / structural environment?
Activity Changes	21	What events / activities need to be sustained?
Participants	22	What are participating elements and what are their responsibilities?
Participants Interactions	23	How do the participating elements collaborate to carry out their responsibilities?
Interaction Diagram	24	How can the relationships within the pattern be visualized?
Consequences	25	What are the effects and side-effects of the pattern?
Pattern Flexibility	26	Which aspects of the solution can be varied independently?
Prototype Information	27	How would a concrete implementation look like?
Known use	28	Where has the pattern already been implemented?
Evaluation		
Evaluative Data	29	What is the empirical background of the pattern?
Evaluative Context	30	What methodology guides the evaluation of a pattern?
Evaluation Method	31	What data gathering methods were used?
Stability	32	Is the pattern stable and self-sustaining? (e.g. Alexander's 'Living patterns')
Coherence	33	Does the pattern integrate with other patterns?
Status	34	How well is the pattern established?
References	35	Where can further explanations be found? (based on in-text citations description)

Table 2 provides a further additional tool, by providing an overview of how the pattern descriptors in Table 1 address the various issues described in this paper. By mapping specific descriptors to specific issues, Table 2 is intended for use as a guideline during the pattern description process.

Table 2: Design issues and pattern descriptors

Issue	Descriptors (#)
Cultural values	Including specific domain characteristics (# 4) helps to understand the invariable part of the context which, if variable in the user environment, may lead to the development of a different pattern.
Design values	Design values are included in the contextual description of a pattern. 'Paradigmatic conditions' (# 3) refer to the pattern author's assumption about learning and what is taken as a benchmark to determine design quality. As for the later descriptors #32 to #34 describe general pattern quality characteristics. They indicate the desired evaluation outcome dimensions (coherence, stability and status of patterns) which need to be addressed in addition to the evaluation of pattern-specific outcomes.
Research values	Descriptors #29 to #31 represent the 'how' of the evaluation or, as outlined in the previous discussion, the 'data and method' perspective on patterns.
Context	Pattern conditions (# 2) describe context as a prohibitive as well as an enabling factor, thereby increasing the transfer of negative and positive design expertise captured in a pattern.
Complexity	Grand theories reduce the complexity of the actual design conditions by imposing a reductionist model of the situation. Whereas a theory uses universal concepts, a pattern reduces complexity by referring to repeating design problems under similar conditions. Though patterns are less generalisable, their format has the potential to better adapt to the complex variety of design situations.
Emergent Complexity	Abstract principles (# 18) and concrete examples (# 27& #28) support the designer in learning and understanding a pattern's meaning. The responsibility for deciding whether a new design condition makes it necessary to alter or discard the pattern is left to the designer. Both, abstract and concrete pattern presentations can provoke a hermeneutic dialogue between the pattern text and the designer. A pattern which provides more examples is likely to allow for more learning if each example represents a slightly different implementation of a general principle. The degree to which a principle is malleable is captured by descriptor #26.
Descriptive Complexity	Patterns combine different modes to present design experiences. This may range from a metaphorical picture (# 20), via textual descriptions of pattern components (#20, #21) to a process diagram (#24). Again, representational gaps due to the increasing complexity of a pattern need to be filled by the reader of the pattern. Such a gap may lead to a new pattern which eventually feeds into the larger pattern (#5) or may be filled with an additional example (#28).

In the 15 pattern related projects we analysed for this paper, we found no other authors who had developed as many descriptors as we have suggested here (Gamma et al., 1995) developed the most detailed pattern format, using 16 descriptors). We are aware that, in suggesting 35 descriptors, we have increased the effort of describing a pattern considerably – but we believe that the lack of effective pattern dissemination to date justifies our approach. Clearly, however, only further studies of the pattern dissemination process by a number of users will show if this additional effort will translate into the wider adoption of educational technology patterns which we anticipate.

Conclusion

Educational designers are faced with the problem of constantly reinventing concepts which may well have already been developed by others. The complexity of course design and the contextual importance of both the educational environment itself and the theoretical paradigms within which educators work makes the reuse of other developers' solutions problematic. One potential solution to this problem is the use of patterns, which provide a guideline and the ability to build on earlier work, without tying the educational designer too tightly into the details of his/her predecessor's activities or theoretical viewpoint. In this paper, we have endeavoured to show that it is the ways in which patterns are described which can have the greatest influence on the later reuse of those patterns. In particular, we have focused on the issues of context and of value systems – building on literature in this area and highlighting the cross-domain issues which both complicate the development and use of patterns in educational design, as well as enhancing the potential flexibility of patterns within this context.

The paper not only investigates the theoretical contribution which earlier researchers into patterns have made to this discussion, but also provides a detailed guide to suggested pattern descriptors – contained within two Tables, for ease of reference and use. We hope that this paper will also serve to draw together some of the many separate threads of research in this field, linking the description of patterns across architecture, software development and educational design.

References

Alexander, C. (1979). *The timeless way of building*. New York: Oxford University Press.

Anderson, E., Plessis, J. d. & Nickel, T. (2001). Participation in international teleconferences and discussions: Implicit assumptions. *Educational Technology, Research and Development,* 49(3), 118–123.

Banathy, B. H. & Jenlink, P. M. (2003). Systems Inquiry and Its Application in Education. In D. H. Jonassen (Ed.), *Handbook of Research for Educational Communications and Technology* (2nd ed., pp. 37–58). Mahwah, NJ: Lawrence Erlbaum Assoc Inc.

Bryant, G. (1994). Christopher Alexander Visits The Oregon Experiment. *RAIN Magazine, XIV*(4), http://www.rainmagazine.com/architecture/oregonex2.html [viewed 15. Jun 2006].

Conole, G., Dyke, M., Oliver, M. & Seale, J. (2004). Mapping pedagogy and tools for effective learning design. *Computers & Education,* 43(1–2), 17–33.

Coplien, J. O. (1997). Idioms and patterns as architectural literature. *Software, IEEE, 14*(1), 36–42.

Coplien, J. O. (2004). Patterns of engineering. *Potentials, IEEE,* 23(2), 4–8.

Coplien, J. O. & Devos, M. (2000). *Architecture as Metaphor.* Paper presented at the World Multiconference on Systemics, Cybernetics and Informatics, Orlando, Florida.

Derntl, M. & Motschnig-Pitrik, R. (2005). The role of structure, patterns, and people in blended learning. *The Internet and Higher Education,* 8(2), 111–130.

Dillenbourg, P. (2002). Over-scripting CSCL: The risks of blending collaborative learning with instructional design. In P. A. Kirschner (Ed.), *Three worlds of CSCL: Can we support CSCL?* (pp. 61–91). Heerlen: Open Universiteit Nederland.

Dreyfus, H. L. (1992). *What computers still can't do: a critique of artificial reason*. Cambridge, Mass.: MIT Press.

Emmeche, C. (1997). Aspects of Complexity in Life and Science *Philosophica, 59*(1), 41–68.

Feyerabend, P. K. (1988). *Against method* (Rev. ed.). London ; New York: Verso.

Fowler, M. (1997). *Analysis patterns: reusable object models*. Menlo Park, Calif.: Addison Wesley.

Gadamer, H.G. (1994). *Truth and method* (2nd rev. ed.). New York: Continuum.

Gamma, E., Helm, R., Johnson, R. & Vlissides, J. (1995). *Design patterns: elements of reusable object-oriented software*. Reading, Mass.: Addison-Wesley.

Goldstein, J. (2004). Why complexity and epistemology? *Emergence: Complexity & Organization,* 6(3), 2–3.

Goodyear, P. (2005). Educational design and networked learning: Patterns, pattern languages and design practice. *Australasian Journal of Educational Technology,* 21(1), 82–101.

Grabow, S. (1983). *Christopher Alexander: the search for a new paradigm in architecture*. Stocksfield; Boston: Oriel Press.

Green, S., Jones, R., Pearson, E. & Gkatzidou, S. (2006a). Accessibility and adaptability of learning objects: responding to metadata, learning patterns and profiles of needs and preferences. *ALT-J,* 14(1), 117–129.
Green, S., Jones, R., Pearson, E. & Gkatzidou, S. (2006b). Learning patterns and learner profiles in learning object design. *ALT-J,* 14(2), 203–207.
Habermas, J. (1971). *Knowledge and human interests* (Shapiro, Trans.). Boston: Beacon Press.
Heath, A. (2006). Exploiting opportunities computers provide to deliver content people can use in context. *ALT-J,* 14(2), 199–202.
Marshall, H. H. (1996). Implications of differentiating and understanding constructivist approaches. *Educational Psychologist,* 31(3/4), 235.
Motschnig-Pitrik, R. & Derntl, M. (2005). *Learning Process Models as Mediators between Didactical Practice and Web Support.* Paper presented at the ER 2005: 24th International Conference on Conceptual Modelling, Klagenfurt, Austria.
Ngwenyama, O. K. & Lee, A. S. (1997). Communication richness in electronic mail: Critical social theory and the contextuality of meaning. *MIS Quarterly,* 21(2), 145.
Niegemann, H. M., Hessel, S. & Domagk, S. (2004). *Pedagogical design patterns for e-learning: A new approach to instructional design theory.* Paper presented at the International Conference on Computers in Education, Melbourne, Australia.
Orlikowski, W. J. & Baroudi, J. J. (1991). Studying Information Technology in Organizations: Research Approaches and Assumptions. *Information Systems Research,* 2(1), 1.
Popper, K. R. (2002/1935). *The logic of scientific discovery*. London: Routledge.
Protzen, J. P. (1978). The Poverty of The Pattern Language. *Design Methods and Theories,* 12(3/4), 191–194.
Rumbaugh, J., Booch, G. & Jacobson, I. (1999). The unified modeling language reference manual. Reading, Mass.: Addison-Wesley.
Salingaros, N. A. (2000). The Structure of Pattern Languages. *Architectural Research Quarterly* 4(2), 149–161.
Savin-Baden, M. (2004). Understanding the impact of assessment on students in problem-based learning. *Innovations in Education & Teaching International,* 41(2), 223–233.
Schein, E. H. (1996). Culture: The Missing Concept in Organization Studies. *Administrative Science Quarterly,* 41(2), 229–240.
Schmidt, D. C. (1995). Using design patterns to develop reusable object-oriented communication software. *Communications of the ACM,* 38(10), 65–74.
Voigt, C. & Swatman, P. M. C. (2006). *From surface to meaningful collaboration: An activity theory perspective.* Paper presented at the International Conference: e-Society 2006 (IADIS), July 13–16, Dublin, Ireland.
Weber, M. (1922). *Economy and Society*. Berkeley: University of California Press.
Wottawa, H. & Thierau, H. (1998). *Lehrbuch Evaluation* (2. Auflage ed.). Bern: Huber.
Yin, R. K. (2003). *Case study research: design and methods* (3. Ed.). Thousand Oaks Calif.: Sage.

Author contact details

Christian Voigt, School of Computer and Information Science, University of South Australia, Adelaide, Australia. Email: christian.voigt@unisa.edu.au.

Paula M.C. Swatman, Professor of Information Systems, School of Computer and Information Science, University of South Australia, Adelaide, Australia. Email: paula.swatman@unisa.edu.au.