IDENTIFYING DESIGN PATTERNS FOR EDUCATION AND TRAINING

Steve Flowers
U.S. Coast Guard
Performance Technology Center
Training Center Yorktown, VA
sflowers@tcyorktown.uscg.mil

Ian Douglas, Patrick Brandt
Learning Systems Institute, Florida State University Tallahassee, Florida
idouglas@lsi.fsu.edu

ABSTRACT

The idea of patterns emerged from architectural design where it was discovered by Christopher Alexander that in successful building and city planning, commonly occurring patterns of design are apparent. A pattern is an abstract representation of a "common problem" and "effective solution" pairing. Patterns are developed by collecting case studies of both successful and unsuccessful designs, abstracting the key features of successful designs, communicating the patterns using a pattern language, and community validation of the patterns. Since its inception in architectural design the general concept of patterns has been adapted and applied to a number of other domains, e.g. software development, where its has been incorporated with the object concept as one of the primary means of facilitating reuse in development. In this presentation we will review attempts to adapt the pattern concept to the design of instruction and performance support. We will examine options for pattern description, and sharing. Experience using computer based training in the Coast Guard is presented as an example of how patterns can become a powerful tool in increasing the quality and perceived value of contemporary training and performance interventions.

ABOUT THE AUTHORS

Steve Flowers is a learning systems engineer at the U.S. Coast Guard's Performance Technology Center onboard Training Center Yorktown, VA.

Ian Douglas is an assistant professor in the College of Information and a faculty member of the interdisciplinary Learning Systems Institute (LSI) at Florida State University.

Patrick Brandt is a graduate research assistant in the Learning Systems Institute (LSI) at Florida State University.

IDENTIFYING DESIGN PATTERNS FOR EDUCATION AND TRAINING

Steve Flowers
U.S. Coast Guard
Performance Technology Center
Training Center Yorktown, VA
sflowers@tcyorktown.uscg.mil

Ian Douglas, Patrick Brandt
Learning Systems Institute, Florida State
University Tallahassee, Florida
idouglas@lsi.fsu.edu

INTRODUCTION: AN OVERVIEW OF PATTERNS

In *The Timeless Way of Building*, Alexander speculates that a detailed and fully integrated hierarchy of architectural design patterns can serve as a guide to the construction of urban environments and individual structures within these environments (Alexander, 1979). He was the first to develop the syntax of a pattern and define how relationships between specific patterns could comprise an entire pattern language.

A pattern is a record of how a particular recurring problem has been successfully solved in the past. It is general enough to be adapted and reused in a way that matches a particular situation. A pattern attempts to provide the best solution to a problem by recognizing and recording principles that are practiced by the best designers. The format of patterns may vary according to the domain to which they are applied. However, every pattern contains these general characteristics: a name that identifies the pattern, a problem statement that specifies the context in which the pattern is applicable, the forces that shape the resulting solution to the stated problem, and the solution itself. Applying this simple format, one of Alexander's own patterns is presented presented in table 1 (Alexander, 1977).

Alexander also specifies that examples be provided (either through photographs or diagrams, in the case of architecture) that help illuminate the solution explained in the pattern. He rates his patterns according to their significance, with some patterns being "more true, more profound, more certain than others" (Alexander, 1979). He used asterisks that were included next to the name of a pattern to indicate its level of significance. Two asterisks indicated a true invariant, a solution that is true for any variation of the related problem statement. One asterisk indicates that the solution can be refined and that there may be a different solution for the given problem. No asterisk indicates that the given solution is merely one of many solutions to the given problem.

Table 1. Example of one of Alexander's patterns

Name:	Master and Apprentices *
Problem:	The fundamental learning situation is one in which a person learns by helping someone who really knows what he/she is doing.
Forces:	…learning from lectures and books is dry as dust…The schools and universities have taken over and abstracted many ways of learning which in earlier times were always closely related to the real work of professionals, tradesmen, artisans, independent scholars…An experiment by Alexander and Goldberg has shown that a class in which one person teaches a small group of others is most likely to be successful in those cases where the "students" are actually helping the "teacher" to do something to solve some problem, which he is working on anyway—not when a subject of abstract or general interest is being taught. (Report to the Muscatine Committee, on experimental course ED. 10X, Department of Architecture, University of California, 1966).
Solution:	Arrange the work in every workgroup, industry, and office, in such a way that work and learning go forward hand in hand. Treat every piece of work as an opportunity for learning. To this end, organize work around a tradition of masters and apprentices: and support this form of social organization with a division of the workplace into spatial clusters—one for each master and his apprentices—where they can meet and work together.

Another element of Alexander's format is a "related patterns" section which records any patterns that provide either more detailed or more general contexts relevant to the problem stated by a particular pattern. Relationships can be defined between patterns that outline a pattern language, providing a network of context-sensitive solutions for the entire domain of problems. For example, the *Master and Apprentices* pattern relates to the higher-level (more general) patterns of *Self-Governing Workshops and Offices*. The *Master and Apprentices* pattern is further developed by the lower-level (more specific) patterns of *Half-Private Offices*, *Workspace Enclosures*, *Common Areas at the Heart*, *Communal Eating*, *Small Work Groups*, and *Small Meeting Rooms*.

The pattern concept has found resonance in several fields since Alexander proposed the first pattern language. These include software engineering (Gamma et al., 1995), interaction design (Bayle et al., 1998), and sociology (Cowan, n.d.). Patterns are not only capable of capturing reusable design knowledge, they may also capture knowledge about optimal approaches in each stage of the development life-cycle. Patterns can also be stored in a digital repository for further reuse where they become the responsibility of communities of practice or organization elements which will continually evalaute their validity through their collective experience.

Analysis Patterns
Analysis patterns (Fowler, 1997) help analysts decide what kind of analysis to use in different situations. These patterns may describe analysis models, techniques, and data collection alternatives that vary according to an organization's size or function (e.g. over 25,000 employees, under 100 employees, for-profit, government). A collection of patterns can effectively support the analysis process by providing a range of alternatives to various scenarios. The use of analysis patterns could be especially valuable by better framing the situation and suggesting alternate approaches.

Design Patterns
Once a solution has been selected during the analysis process, patterns that support the selected solution can guide the specific design of the solution, e.g. software design patterns. These solution-specific patterns may have already been developed by others in the design pattern community; if not, a pattern writer could study other instances of the prescribed system and record his/her own collection of design patterns. For example, if the solution for a given problem recommended the use of a Learning Management System (LMS), a set of patterns particular to LMS design could be used to manage the construction of the LMS.

Implementation Patterns
Once a set of design patterns that guide the construction of a particular type of solution has been identified, another set could recommend the ways in which the system would be most useful in any given context. For example, this pattern collection may include patterns related to change management or diffusion of innovation. Implementation patterns may include solutions to specific technical specifications or cultural barriers unique to an organization or organizational component.

Evaluation Patterns
After a system has been implemented within a particular performance-related context, the effectiveness of the system must be judged. The methods of rating a system's impact on performance can be recommended through the use of evaluation-based patterns. These patterns would prescribe certain metrics that could be used to rate performance and describe how these metrics could be measured. In a well organized pattern system, the evaluation of a whole solution would provide feedback for the pattern components that comprise the solution.

Although the pattern concept has general relavance to any domain of problem solving it is only relatively recently that the concept has begun to be applied to the domain of education and training.

PATTERNS IN THE DOMAIN OF PERFORMANCE AND LEARNING

The intentional disciplined use of patterns is not common among the majority in the performance technology, training and education fields. One example of the application of patterns in the learning domain relates to the design of learning management systems (Avgeriou et al., 2003;). LMS design patterns are used to manage the construction of a learning management application. The LMS pattern format proposed by Avgeriou et al. identifies the elements that shape the solution statement in a "Motivation" section and adds two new categories to the basic four first proposed by Alexander: "User Category," and "Known Uses." The "User Category" field identifies the particular types of users that are most affected by the problem identified in the pattern. "Known Uses" indicates which LMS applications already exhibit the qualities expressed in a pattern. Unlike Alexander's pattern definition, there is no method associated with the pattern for rating its quality. Avgeriou, et al. (2003)

have created nine patterns specific to LMS design (see example in table 2).

Table 2. Example of an LMS pattern

Name:	Synchronous collaborative learning
Problem:	Learners and instructors must be able to collaborate and cooperate with each other and with their peers in a synchronous manner
Motivation:	Synchronous collaborative learning is a computer-mediated effort that simulates face-to-face interaction. Since body language and facial expressions cannot be conveyed through asynchronous communication, real-time communication allows contributions participation, sharing information and social dialogue at a distributed environment. The main advantages of synchronous multimedia communication are: •"Next best thing to being present at a lecture hall" •Visual medium; students and teachers can relate to one another •Good for distance education novices to develop "learning community"
Solution:	Synchronous multimedia communication tools make it possible for learners and instructors at different sites to partake in the same conference at the same time through the "magic" of two-way audio and two-way compressed video. Examples of types of synchronous communication include: • text-based Internet chats • instant messaging • audio and video conferencing • virtual whiteboard applications • shared applications
User Category:	Learners and instructors
Known Uses:	All LMS provide some sort of chat or conferencing service.
Related Patterns:	Asynchronous Collaborative Learning, Student Group Management, Student Assignments Management

Performance Improvement Patterns

Performance improvement patterns address the class of problems that identify gaps between the ideal performance and actual performance of workers (see example in table 3). These patterns suggest processes or techniques that can be used to improve worker performance. For example, consider a retail organization whose sales associates are having difficulty handling inventory changes. Existing methods of referencing current inventories are either quickly outdated (in the case of paper-based references) or difficult to learn (in the case of computer-based references). In this example, the problem and forces that shape the solution have been identified. Once a solution is implemented that successfully closes the performance gap, it can be recorded along with the problem and the forces in a performance improvement pattern that can be reused whenever another organization is confronting a similar problem.

Table 3. Example of a Performance pattern

Name:	Forgetting sequence of steps provided in training.
Problem:	People are taught to perform a sequence of actions on some artifact (e.g. piece of equipment) in training and many frequently forget this sequence when asked to perform on the job.
Motivation:	Most equipment requires people performing certain roles (operator, maintainer, troubleshooter) to achieve a particular goal, e.g. setting up a radio to operate in frequency hopping mode. These goals often involve following a set of five to fifteen steps. If the sequence is not correct or a step is missed, the goal will not be achieved.
Solution:	Provide a quick reference guide, at the point of performance. For example, on a laminated panel on the equipment itself or the equipment packaging. If this is not possible, provide a quick reference sheet on card or on a PDA.

User Category:	Equipment users, equipment procurement
Known Uses:	Many known uses, including automotive, heavy equipment, computer equipment. Can be used for setup, diagnosis, shutdown, maintenance, and operation.

THE RELATIONSHIP BETWEEN PATTERNS AND GUIDELINES

Guidelines represent shared standards for those involved in the design and development of any product. However, they often require fairly sophisticated interpretation by developers in order to be useful. It is also difficult to discover the problem that the guideline is actually trying to address, since it is not explicitly stated (Welie, et al., 2000). Difficulty interpreting guidelines, the effort required to find relevant guidelines, and their simplistic nature combine to discourage their use. Guidelines are developed by organizations or individuals in order to capture both the philosophy behind the design of their systems and how specific design issues should be addressed. For example, one of IBM's web design guidelines prescribes the following guideline for web accessibility in digital content (http://www-306.ibm.com/ibm/easy/eou_ext.nsf/Publish/748):
"Ensure that image maps are accessible to vision-impaired users." Despite the useful advice conveyed in by this guideline, no concrete details are provided as to how an interface actually addresses it. Image maps can be used in several different ways; image maps that are used as a method of website navigation will have to provide different information from image maps that are used for other reasons. What type of textual information is supposed to be provided to vision-impaired users when they are confronted with an image map? With no context provided, this question cannot be answered. What is needed is a method of providing solutions for problems that can be applied in context to specific types of software applications. The pattern concept is an obvious solution to the problem of providing context to guidelines.

THE PATTERN DEVELOPMENT PROCESS

Writing patterns requires that one recognize and record common problems and matching solutions inherent to a particular domain. Most patterns are created by individuals or groups of people who are experts or practitioners in a particular field. For example, Software Engineering design patterns are typically created by programmers in workshops (Dearden, et al., 2002) like those that take place during the *Pattern Languages of Programming* conference (http://hillside.net/plop/2004/). Pattern creation by committee can be facilitated by the pattern collection for writing workshops created by Jim Coplien and Bobby Woolf (Coplien & Woolf, 1999). Though experts and practitioners are presumably those most qualified to recognize patterns in their domains of expertise, pattern creation is not necessarily restricted to those who are experts in their field (Alexander, 1979). Patterns can be created by anyone who takes the time to discover and record them. Techniques for pattern writing and pattern language construction have been documented, appropriately, in a pattern language for writing patterns (Meszaros & Doble, n.d.).

Though patterns are primarily considered problem-solving tools, they have other applications beyond problem solving. They have a didactic role in that novice designers are able to learn some of the problem-solving techniques used by experts in their field (Tidwell, n.d.). Those who have no knowledge of a particular domain can gain insight into it simply by looking at the domain's relevant patterns. To this end, patterns may be used to facilitate the participation of all stakeholders in the design process who have a vested interest in the outcome. For example, Alexander's patterns were used to facilitate the exchange of ideas between architects, doctors, and staff during the design of a psychiatric clinic in California. His pattern language became the medium "in which people worked out their disagreements, and in which they built a common picture of the building and the institution as a whole" (Alexander, 1979).

Rating the effectiveness of a pattern outside of Alexander's collection is, at this point, a fairly abstract science. Alexander's quantitative system of ratings is not found in other disciplines where patterns have been applied. For example, criteria have been established for judging the quality of a software engineering design pattern, but the determining factor for evaluating these patterns exists in their review and acceptance by "qualified peers" (http://www.antipatterns.com/whatisapattern); this could occur either by workshop discussion or via criticism in a public forum such as an on-line discussion board. A litmus test that can determine if a pattern is useful or not is whether it has been used by someone other than its author (Vlissides, 1996); however, the number of times a pattern is copied should not be the sole manner by which the quality of a pattern is evaluated. The defining factor in determining a pattern's quality is the degree to which

it reinforces the principles of the domain for which it is created. Thus, determining the quality of a pattern is a rather subjective endeavor. If a pattern is recording a solution that is generally considered the best (or at least most appropriate) for a given problem, the pattern will be useful.

As patterns for a given field continue to be created and validated by domain practitioners, relationships can be defined between patterns that outline a pattern language, providing a network of context-sensitive solutions for the entire domain of problems. This network of patterns forms a framework for constructing something that is "functionally and structurally coherent" (Salingros, 2000) for a particular sphere of design problems. Additionally, a collection of patterns for a particular area of practice may be useful without the explicit relationships between patterns that define a pattern language. Gamma's software engineering patterns are organized not so much a pattern language as they are a pattern catalog (Dearden, et al, 2002). Though they may contain cross-references to each other, they do not start from a specific root and carry explicit relationships that tie each pattern into a networked organization; they are organized more like a dictionary where reference can be made when problems arise.

Alexander recommends that his pattern language serve as a "base map" from which one can make their own language by selecting a subset of his patterns that are most useful to them (Alexander, 1977). He states that the patterns in his language may be further refined and adapted (Alexander, 1977), but provides no means for the expansion of his language. However, two basic models of pattern language creation have been defined elsewhere. The first approach is to identify patterns that solve the most atomic design problems for a particular domain, discover the general problem that these groups of atomic patterns solve, and then generalize the patterns identified in these groups to create a pattern of larger scope (Salingros, 2000). The second approach is to identify the most general patterns first and then distil from them more detailed patterns that complete the generalized solutions (Meszaros & Doble, n.d.). Regardless of how patterns are eventually organized in relation to each other, all patterns should be recorded as they are observed and then sorted out later (Salingros, 2000).

A PATTERN SERIES DERIVED FROM THE EXPERIENCE OF THE U.S. COAST GUARD

In 1998, the U.S. Coast Guard Training Center in Petaluma California recognized an alarming rate of poor student performance on final performance assessments for a unit within their electronics technician school. The course staff did not have the resources to restructure the curriculum and extend the time spent within this unit of the school.

The Instructional Support Team (a staff performance services group) was brought in to evaluate the problem and devise a solution. The team discovered that the students were not fully engaged in the learning experience, and in most cases lacked confidence and motivation to successfully complete practice and final performance evaluations. The team guessed that the student's engagement was directly proportional to the transfer of concepts and information, and that sufficient practice would increase student confidence when performing in practical evaluations. The team also guessed that a self-directed experience had the potential to engage, motivate and increase the students's confidence in their own abilities. If this could be accomplished, student retention of the concepts would increase and the reversion rate would decrease.

A supplementary technology solution was recommended to enable students to practice troubleshooting procedures with real feedback in varied situations. The team created a self-directed part task simulation that represented all of the cues and feedback a student would see in the real world situation. The solution addressed problem-solving contexts and provided real feedback for every action and decision. This part task self-directed trainer enabled all students to make mistakes and repeat the performance as many times as needed. When the supplementary simulation was rolled out a few weeks later, the measurable benefits included enhanced concept transfer and an apparent increase in student confidence. The end result included higher final performance evaluation scores and consequently a lower rate of reversion over the following months.

This experience represented a complex set of problem solution pairs that represent a successful design experience. Patterns are way in which this experience can be abstracted from the specific situation into a form that can be shared with other instructional designers and performance technologists. Experience is important and capability through experience is the desired end-state. In the best form of experience the organization is resourced to provide, or is able to motivate the performer to provide themselves.

Today, the Coast Guard Performance Technology Center in Yorktown, VA sees pattern study and the application of patterns as a potential way to overcome

the typical deficiencies of self-directed learning events. There is a high level of dissatisfaction with e-learning courses and self-directed learning events. A recent study (DDI Survey, reported by Van Buren, ASTD 2003) revealed that of 139 companies in 15 countries, 75 percent rated the effectiveness of eLearning as less than 5 on a scale of 1 to 10 (average is 3.9). Based on the Coast Guard's experience, this number represents a vastly different audience than a military and action performance oriented organization. Despite the difference in audience type, many of the same factors apply. Factors like culture and generation dominate as problems for rolling out self-directed learning experiences. The most important ingredients to the success of the medium, in our opinion, are authentic context, experiential activity and apparent excellence. Without these, eLearning courses will be ineffective and de-motivating. Which, coincidentally, are attributes common to many courses, particularly online ones.

The U.S. Coast Guard has just begun implementing, building, and testing the pattern concept. Development and evaluation of new patterns is underway in order to confirm their anticipated efficacy in capturing and sharing design knowledge inherent in solutions with proven effectiveness. This is the nature and ultimate intent of any pattern system. Table 4 shows a pattern derived from the experience of projects like the one described above. This is a high level pattern from which more specific patterns can be derived.

Patterns can be built and applied to every category of performance and seem to be an excellent way to explicate tacit knowledge and share it among designers. The Coast Guard's experience and years of successful human performance technology practice suggest both that pattern application is a good fit for the Coast Guard Performance Solutions group, and that this group will be successful in the continued application and evaluation of a pattern system.

Table 4. A Coast Guard learning motivation guideline, expressed in the form of a pattern language

Name:	Motivate and involve self-directed learners.
Problem:	Students using a self-directed learning system often lack the motivation to complete their instruction.
Motivation:	Uninspired students will typically not finish self-directed instruction. Unfinished instruction is typically less effective than finished instruction. When knowledge is presented in traditional ways, e.g. page turning CBT, PowerPoint, it can often create a poor experience for participants. This is particularly so in relation to CBT as observed by numerous studies of mediocre courseware. The transfer of information in a presentation, void of context, contributes to the question 'why am I learning this?'. If learning is divorced from the student's performance expectations following the completion of the learning, satisfaction and retention likely will suffer greatly.
Solution:	Design learning presentations and activities in the context of the performance and pattern after successful and popular media. Apply successful scenario format, derived from historical or fictional narratives. Evolve patterns based on evaluation of implemented solutions. When games are used, learning objectives and performance context must be directly linked to the steps required for the successful completion of the game scenario.
User Category:	Learners, instructors and designer / developers, managers

THE ADVANTAGES OF KNOWLEDGE REUSE IN PATTERN APPLICATION

ADL SCORM is the most prominent initiative in terms of reuse in the military, and will eventually pay the dividends promised by the standard. However, the Coast Guard PTC recognizes that between the deficiencies in the current standards implementation, the lack of relevant content, and the inability to intelligently combine SCO's of different contexts, the best reuse application focus, for Coast Guard purposes, is on the elements that attribute to a completed content object.

If it makes sense to reuse the end result (the SCORM object), it also makes sense to reuse and share the

analysis and design knowledge that led to that end result. Given that different services may like to see different terminology, colors, pictures, and context in the end result, it may be more feasible to share of the abstracted design experience through patterns and common or similar design constructs. This is certainly been the case in the software industry, where patterns are currently seen as facilitating the reuse that objects failed to deliver to the extent anticipated.

What we all really want is to take advantage of the work it took to get to the end state. This includes the design, the media and all of its constituent elements. Because we are really evaluating the aggregate of all of these components as they apply to the accomplishment of improved performance in the military. Patterns are a way for us to begin to share and improve the problem solving portions of this work.

Evaluation of a monolithic solution set is folly. Few organization elements even attempt to evaluate their solutions on a continual basis and of those that do a slim minority do it successfully. What if we broke these solutions down and directly associated them with problems they are solving? What if we watched for the recurrence of these problems, the creation of new, related, problems, and the degree of recurrence? It becomes much easier to evaluate a solution when we can wrap our arms around it.

A CHALLENGE TO ACADEMIA, INDUSTRY, AND GOVERNMENT

In order to progress patterns within the government and military community, it is necessary to establish a pattern study group, a military standard method for capturing, reviewing and rating patterns, and mechanisms for verifying and sharing patterns. Once such a library of patterns is established and supported by experienced practitioners across the government services, contractors can be required to apply the solution catalog to the products they develop. The resource savings potential is staggering if one considers that the most expensive portion of eLearning development is the design of the intervention. The potential to save resources is even more staggering if one considers the number of failed or poor solutions that have been bought and paid for with government eLearning funds. How can government and industry work together to advance the field and improve future products? Perhaps by applying the patterns concept to learning systems design and human performance support, the industry and field can evolve together.

REFERENCES

Alexander, C. (1979). *The Timeless Way of Building*, New York: Oxford University Press.

Alexander, C. (1977). *A Pattern Language*, New York: Oxford University Press.

Bayle, E, et al. (1998). *Putting at all Together: Towards a Pattern Language for Interaction Design. SIGCHI Bulletin, 30*(1). Retrieved June 6, 2004, from www.acm.org/sigchi/bulletin/1998.1/erickson.html.

Coplien, J., & Woolf, B. (1999). *A Pattern Language for Writers' Workshops.* Retrieved June 10, 2004 from http://www.bell-labs.com/user/cope/Patterns/WritersWorkshops/.

Cowan, S., et al. (n.d.). *What Does a Sustainable Society Look Like?* Retrieved June 4, 2004, from http://www.conservationeconomy.net/.

Dearden, A., Finlay, J., Allgar E. & McManus, B. (2002). Using Pattern Languages in Participatory Design. In Binder, Gregory, & Wagner (Eds.), *PDC 2002, Proceedings of the Participatory Design Conference* (104-113). Palo Alto, California. Retrieved May 30, 2004, from http://www.lmu.ac.uk/ies/comp/research/isle/patterns/papers/PDCpaper.pdf.

Fowler, M. (1997). *Analysis Patterns:Reusable Object Models,* Menlo Park: Addison-Wesley.

Gamma, E., et al. (1995). *Design Patterns: Elements of Reusable Object-Oriented Software,* Boston: Addison-Wesley.

Avgeriou, et al. (2003). *Towards a Pattern Language for Learning Management Systems*. *Educational Technology & Society, 6*(2). Retrieved February 16, 2004, from http://ifets.ieee.org/periodical/6-2/2.html

Richard N Griffiths and Lyn Pemberton *"Don't Write Guidelines Write Patterns!"* http://www.it.bton.ac.uk/staff/lp22/guidelinesdraft.html

Meszaros, G., & Doble, J. (n.d.). *A Pattern Language for Pattern Writing*. Retrieved June 10, 2004, from http://webclass.cqu.edu.au/Patterns/Resources/writers/language/patterns.html.

Salingaros, N. A. (2000). The Structure of Pattern Languages. *Architectural Research Quarterly, 4,* 146-161. Retrieved June 10, 2004, from www.math.utsa.edu/sphere/salingar/StructurePattern.html.

Tidwell, J. (n.d.). *Common Ground: A Pattern Language for Human-Computer Interface Design*. Retrieved June 1, 2004 from http://www.mit.edu/~jtidwell/common_ground.html.

Vlissides, J (1996). *Seven Habits of Successful Pattern Writers*.
Retrieved June 6, 2004 from http://hillside.net/patterns/papers/7habits.html.

Welie, M., Veer, G.C., & Eliëns, A. (2000). Patterns as Tools for User Interface Design. In *International Workshop on Tools for Working with Guidelines* (313-324). Biarritz, France. Retrieved June 6, 2004, from http://www.cs.vu.nl/~martijn/publications_all.html.