

Pattern Archive: A cross section of patterns forms

Thursday, February 6, 2003

Dale Evernden

IART 438

Faculty Advisor: Ron Wakary

[Articles and papers On writing patterns](#)

[A typical Alexander Pattern](#)

Pattern Archive:

[Software patterns](#)

- *EPISODES: A Pattern Language of Competitive Development*
- *Rappel Pattern Language.*

[Interaction Design patterns](#)

- *Experiences -- A Pattern Language for User Interface Design*
- *Interaction Patterns in User Interfaces*

[Business Patterns](#)

- *Coplien—Organizational Patterns*
- *Business process reengineering*
- *Risk management pattern catalogue*

[Telecommunication Patterns](#)

- *Fault tolerant telecommunication pattern system*

[Pedagogical Patterns](#)

- *Fourteen pedagogical patterns (Bergen)*
- *A presentation pattern language*

[Game design Patterns](#)

- *Kreimeier game patterns*
- *Multiplayer game Design Patterns*

[Ecology Design Patterns](#)

- *Patterns of a Conservation Economy*
- *Ecopatterns—a pattern language for ecosystems*

[HCI Patterns](#)

- *Social Issues and Software Architecture*

Appendices

Appendix A

Appendix B

Appendix C

Appendix D

Appendix E

Appendix F

Appendix G

On Writing Patterns

[Seven Habits of Successful Pattern Writers](#), by John Vlissides

[Patterns: The Top Ten Misconceptions](#), by John Vlissides

[A Pattern Language for Pattern Writing](#), by Gerard Meszaros and Jim Doble

The Alexander Pattern Form

There are [many variations](#) on Alexander's [3] original definition of pattern, but the main elements are these, as illustrated with a superb example from Alexander.

Name: A name for the pattern

Example: Window Place

Context: A context for the design problem

Example: Design of a residential room

Forces: Forces which require resolution

Example: People want to sit and also be in daylight.

Problem: A problem growing from the forces

Example: If all seating is away from the windows, then these forces are not resolved, and people will always be dissatisfied in one way or the other.

Solution: A known solution, proven in practice

Example: Build seating into the window -- the traditional window seat.

A **pattern language** is [a collection of patterns](#) that can solve all the problems in a particular domain. It may include a method for connecting patterns into whole "architectures" for the domain. (Less ambitiously, a "pattern system" [5] covers only parts of a domain.)

Archive

Software patterns:

Title: *EPISODES: A Pattern Language of Competitive Development*

Area of application: Software development

Form: Portland

URL: <http://www.bell-labs.com/cgi-user/OrgPatterns/OrgPatterns?EpisodesPatternLanguage>

Abstract:

- This pattern language describes a form of software development appropriate for an entrepreneurial organization.
- These patterns tell what decisions can be made, in fact should be made, to maintain continuous forward motion through iterative development.

- The language addresses a wide variety of development issues. These have been organized into topic areas that could be described as top-down or chronological.

Note: Explaining title--we are particularly interested in the sequence of mental states that lead to important decisions. We call the sequence an episode. An episode builds toward a climax where the decision is made.

Title: *Rappel Pattern Language*.

Area of application: Requirements analysis for object oriented software design.

Form: “problem, discussion, solution”

URL: <http://www.bell-labs.com/cgi-user/OrgPatterns/OrgPatterns?RappelPatternLanguage>

Example: Managing and Meeting Customer Expectations

<http://www.bell-labs.com/cgi-user/OrgPatterns/OrgPatterns?CustomerExpectations>

Abstract:

The goal of this language is:

- To provide a set of techniques and methods that will lead to a more thorough analysis and understanding of a problem area
- To provide a framework for effectively capturing requirements so that a software product can be evaluated, designed, built and tested
- To be able to trace the design of the system back to the original business and system objectives

Interaction Design Patterns:

Title: *Experiences -- A Pattern Language for User Interface Design*

Area of application: software design

Form: Alexanderian

URL:

www.maplefish.com/todd/papers/experiences/Experiences.html#Interaction%20Style

Example:

www.maplefish.com/todd/papers/experiences/Experiences.html#Single%20Setting

Abstract: By using the patterns described here, you should be able to develop languages that help you build environments that will be pleasurable and productive to use. You won't find information here on how to use icons, pop-up menus, dialog boxes and other interface gadgets. Our primary focus is on the higher level patterns found in all good user interfaces: Patterns that help us design interfaces that provide the user with positive experiences using well engineered software systems.

Title: *Interaction Patterns in User Interfaces*

Area of application: see title

URL: <http://www.cs.vu.nl/~martijn/patterns/PLoP2k-Welie.pdf>

Form: problem, usability principle, context, forces, solutions, rationale, examples, known uses, counter example.

Example: See paper

Abstract: These patterns are focused on solutions to problems end-users have when interacting with systems. The patterns take an end-user perspective, which leads to a format where usability is the essential design quality.

Business Patterns

Title: *Coplien—Organizational Patterns*

Area of application: Organization management

Form: Alexanderian

URL: <http://www.bell-labs.com/cgi-user/OrgPatterns/OrgPatterns?CoplienOrganizationPatterns>

Example: <http://www.bell-labs.com/cgi-user/OrgPatterns/OrgPatterns?FormFollowsFunction>

Abstract:

- This is a family of patterns that can be used to shape a new organization and its development processes.
- It addresses recurring patterns of interaction in organizations, and takes note of recurring patterns that occur between those patterns.
- The patterns presented combine empirical observations with a rationale that explains them.
- These patterns are drawn from peculiar organizations with peculiarly high productivity. The patterns describe practices much different from those found in most project management texts.

Title: *Business process reengineering*

Area of application: developing hyper-productive and adaptable companies that simultaneously provide work environments that increase the quality of life and comfort of their employees.

URL: <http://www.bell-labs.com/cgi-user/OrgPatterns/OrgPatterns?BPRPatternLanguage>

Form: closely resembles the Canonical Form

Example: URL: www.bell-labs.com/cgi-user/OrgPatterns/OrgPatterns?LeaderLeader

Abstract: The BPR pattern language addresses how an organization should evolve over time. Issues addressed are structure, values, processes for embracing enterprise technology.

Title: *Risk management pattern catalogue*

Area of application: Project management

URL: <http://members.aol.com/acockburn/riskcata/risktoc.htm>

Form: [see appendix G](#)

Example: [Team per task](#)

Abstract: Some project leaders seem consistently able to make projects succeed, but they are typically unable to say how in a way that passes on the key information to other project leaders. This catalogue of patterns is an attempt at meeting this need.

Telecommunication Patterns:

Title: *Fault tolerant telecommunication pattern system*

Area of application: Telecommunication

URL: http://www1.bell-labs.com/user/cope/Patterns/PLoP95_telecom.html

Form: Problem, context, solution, forces, resulting context, rationale

Example: People know best ([appendix F](#))

Abstract: These patterns form part of a much larger pattern catalogue in use at AT&T. The patterns presented here form a small partial pattern language within the larger collection of patterns. We chose them because of their interconnectedness, the diversity

of their authorship, and because they are probably well-known to the telecommunications programming community. Many of these patterns work in other domains, but for now, we take telecommunications designers as our audience.

Pedagogical Patterns:

Title: *Fourteen pedagogical patterns (Bergen)*

Area of Application: computer science course development

Form: [click here to go to appendix D](#)

URL: <http://csis.pace.edu/~bergin/PedPat1.3.html>

Example: [Early bird](#)

Abstract: The patterns are not all at the same level of scale. Some speak to the overall course organization and some to very low level things. The general flow is from large structure (semester courses) to small scale (daily activities). A long term goal is to develop them into a proper language. This will require supplementing them with others as well

Title: *A presentation pattern language*

Area of application: Designing and giving presentations

Form: [See appendix E](#)

URL: [Download PDF file here](#)

Example: **Motivation** [See appendix E](#)

Abstract: Giving a good presentation is not easy. It takes a lot of discipline and creativity to prepare and give a presentation that, on the one hand, is received positively by its audience and, on the other hand, has the effects desired by the presenter. This paper gives a handful of recommendations that aid in creating a good presentation. These recommendations are put in **pattern** form and combined into a presentation **pattern language**

Game Design Patterns

Title *Kreimeier game patterns*

Area Of application: Content development for games

Form: “problem, solution, consequences, examples, references.”

URL: http://www.gamasutra.com/php-bin/login.php3?from=/features/20020313/kreimeier_01.htm

Example: [Click here to go to appendix A](#)

Abstract: The game design pattern method proposed here is concerned with content patterns, as opposed to software engineering patterns [19], specializations of which that have been proposed for game programming [33,20].

Title: *Multiplayer game Design Patterns*

Area of application: Multiplayer game design

Form: “problem, solution, consequence, examples, discussion, comments, other names”

URL:

http://www.abc.se/~m10383/Haven/General/Multiplayer_Design_Patterns.html#Format

Example: [click here to go to appendix B](#)

Abstract: this document is a collection of patterns commonly used in multiplayer games. It is not about patterns specific to multiplayer games but the way the patterns are approached and described is focused on multiplayer designs.

HCI Patterns

Title: *Social Issues and Software Architecture*

Area of application: HCI

URL: [click here](#)

Form: very unique ([click here to go to appendix C](#))

Example: “Variation behind interface” [click here](#)

Abstract: N/A – this pattern was included in this list of example based on it’s unique form.

Ecology Patterns

Title: *Patterns of a Conservation Economy*

Area of Application: building ecologically restorative, socially just, and reliably prosperous societies.

Form: Closest to Alexandrian

URL: <http://www.conservationeconomy.net/INDEX.CFM>

Example: “Access to knowledge” ([click here](#))

Abstract: “On this site, fifty-seven patterns provide a framework for an ecologically restorative, socially just, and reliably prosperous society. They are adaptable to local ecosystems and cultures, yet universal in their applicability.”

Note: This site provides a [cool little map](#) that gives the reader a meta perspective of the language as a whole—makes for a better point of entry.

Title: *Ecopatterns—a pattern language for ecosystems*

Area of application: Ecological design

Form: [see template here](#)

URL: <http://www.designmatrix.com/pl/ecopl/index.html>

Example: [Garbage separation at the source](#)

Abstract: This work is based on the *Ecopatterns* course, taught by Gary Swift and Ken Asplund at the School of Design, California Institute of the Arts, in 1973, where the pattern language was applied to ecological design problems.

Appendix A ([back to game design](#))

Paper-Rock-Scissors

Problem: Avoid a dominant strategy that makes player decisions a trivial choice.

Solution: Introduce nontransitive relationships within a set of alternatives, as in the game of paper-rock-scissors.

Consequence: The player is no longer able to find a single strategy that will be optimal in all situations and under all circumstances. She has to revisit her decisions, and, depending on the constraints imposed by the game, adjust to changing situations, or suffer the consequences of an earlier decision.

Examples: The example given by Andrew Rollings is the set of warrior-barbarian-archer from the Dave and Barry Murray game *The Ancient Art of War* (Broderbund 1984). He also describes *Quake's* weapon/monster relations in similar terms: Nailgun beats shambler, shambler beats rocket launcher, rocket launcher beats zombie, zombie beats nailgun [28].

References: Chris Crawford (see "Triangularity" in [15]) provided the first explicit description of the use of nontransitive relationships. Andrew Rollings' discussion of examples uses game theory including detailed payoff, as well as informal fictional designer dialogs.

Appendix B ([back to game design](#))

Multiplayer Game design pattern example

Category—Narrative

Title: Receipt

Problem: To allow players to make progress in a storyline

Solution: Introduce some sort of milestone or landmark action representing the essence of the achievement.

Consequence: In the eyes of the player, the receipt will represent progress.

Examples: If a game has progress, it is either a continuous scale or discrete events. Sometimes discreet events are visible to the player, such as a response in a MUD or MMORPG NPC dialog.

Consider the following MMORPG dialogue between player ("Neo") and NPC ("Child"):

Child says,

 Instead, only try to realize the [truth].

Neo says,

 what truth

Child says,

 There is [no spoon].

Neo says,

 no spon

Neo says,

 what no spoon

Child says,

 There is only your self.

In this dialogue, the action that the player needs to perform to advance is clearly marked with square brackets. In some games you may see these called triggers, since they trigger an event in the game world.

The Receipt usage in this dialogue is the response to the character's repetition of the key phrases ("truth", "no spoon") - these responses are examples of visible receipts. They let the player know that he is making progress. Upon hearing the word "truth", the NPC responds by revealing new keywords.

Comment: This is a special case of the Milestone Pattern. See also the Requirement Pattern to which this is the companion pattern. The Receipt Pattern is not specific to multiplayer scenarios.

Appendix C: [\(back to social issues pattern\)](#)

Form --- Social Issues and Software Architecture

Two templates are used, each showing a common pattern of forces with a useful resolution. A principle is a reaction to a problem or force, in which a design force and its counterforce are declared. A design decision is a reaction to a set of forces, in which a balance point is declared. The principles may be found and used on many projects.

Principles are written as:

Intent: The intended benefit of the pattern;

Force: An external force acting on the project or design;

Principle: A driving force, from a freely chosen principle;

Counter: A counter and limit to the principle. (In each pattern, the counterforce is that too much of a good thing is not a good thing. Adding interfaces and subsystems makes the system, slower, and eventually, harder to understand.)

Design decisions are written as:

Intent: The intended benefit of the pattern

Context: The situation in which the decision takes place;

Forces: What is pulling the designer in various directions;

Resolution: A suitable resolution for these forces in this context.

Appendix D [\(back to 14 pedagogical patterns\)](#)

Pedagogical pattern format

From the feedback acquired from participants at the pedagogical patterns workshops and those who have provided feedback in other ways, a new format for the patterns has been drafted and is open for review.

This format contains the following sections:

NAME: pattern name

DATE: date of last update

AUTHOR: name of person submitting the pattern

THUMBNAIL: short description (abstract) of the pattern

PROBLEM / ISSUE: problem, challenge, or issue that the pattern is addressing

AUDIENCE / CONTEXT: For what type of learners, in what context, is this pattern appropriate?

FORCES: What makes the problem a problem?

SOLUTION: the solution this pattern proposes to the problem

DISCUSSION: resulting content/consequences and implementation issues

SPECIAL RESOURCES: resources needed to use this pattern (things that are not ordinarily available to the person using the pattern)

CONTRAINDICATIONS: when not to use the pattern, including any cultural dependencies

RELATED PATTERNS: The author may want to peruse the web page (and other sources) and comment on any existing patterns related to this one.

EXAMPLE INSTANCES: specific uses of the pattern (who, how, etc.)

REFERENCES / ACKNOWLEDGEMENTS: any citations and/or individuals who should be acknowledged as contributing to this pattern

Appendix E (back to presentation patterns)

Presentation Pattern Example

Title: Motivation

Problem

How can you rouse the audience's interest in your presentation?

Context

You are at the beginning of your presentation and already have the audience's attention.

Forces

* In the very beginning the interest of the audience is easiest to rouse because of their natural curiosity. Later on it becomes more difficult.

* The audience may be sceptical if the content of your presentation could be interesting.

Solution

Motivate *why* you are giving this presentation. Stress the point why the audience should be interested in your presentation.

For instance, ask a thought-provoking question that leads to your subject, or give an EXAMPLE of a pressing problem you are going to deal with in your presentation. If appropriate, tell the audience about your objective.

Resulting Context

You have the audience's attention and interest. Now, if necessary, you should provide the basis for following the presentation by supplying an OVERVIEW and a KNOWLEDGE BASELINE

.

Rationale

Interest creates attention that lasts until the end of your presentation. In contrast, an ICEBREAKER rouses the audience's attention only temporarily.

Example

The presenter continues, "But now your dangerous and laborious days are over. For now Can-Guru's superb new Can-O-Pna is available. Let me tell you more about it."

Related Patterns

MOTIVATION is often tightly coupled with ICEBREAKER. See the Related Patterns section of

Appendix F ([back to telecommunication patterns](#))

Pattern: People Know Best

Problem: How do you balance automation with human authority and responsibility?

Context: High-reliability continuous-running systems, where the system itself tries to recover from all error conditions.

Forces: People have a good subjective sense of the passage of time, and how it relates to the probability of a serious failure, or how it will be perceived by the customer.

- The system is set up to recover from failure cases. (Minimize Human Intervention)
- People feel a need to intervene.
- Most system errors can be traced to human error.

Solution: Assume that people know best, particularly the maintenance folks. Design the system to allow knowledgeable users to override the automatic controls.

Example: As you escalate through the 64 states of Processor Configuration (Try All Hardware Combos), a human who understands what's going on can intervene and stop it.

Resulting Context: People feel empowered; however, they also are responsible for their actions. This is an absolute rule: people feel a need to intervene. There is no perfect solution for this problem, and the pattern cannot resolve all the forces well. Fool Me Once is a partial solution, in that it doesn't give the human a chance to intervene.

Rationale: There is no try; there is only do or fail-Yoda, in Star Wars.

Consider the input command to unconditionally restore a unit. What does "unconditional" mean? Let's say that the system thinks that the unit is powered down; what should happen when the operator asks for the unit to be restored unconditionally? Answer: try to restore it anyhow, no excuses allowed; the fault detection hardware can always detect the powered-down condition and generate an interrupt for the unit out of service. Why might the operator want to do this? Because it may be a problem not with the power, but with the sensor that wrongly reports the power is off.

Notice the tension between this pattern and Minimize Human Intervention.

Author: Robert Gamoke, 1995/03/24

Appendix G ([back to risk management patterns](#))

Risk management form description

A risk management catalog should give both diagnosis aid and prescription. Therefore, I am suggesting here the following template for these risk management entries:

- *Name* - The name of the pattern, and person nominating it.
- *Chapter* - The primary and secondary issues addressed.
- *Sensation* - What you might be feeling like at this time
- *Symptoms* - Relevant characteristics of the project at this time

- *Forces* - Forces pushing you in particular directions
- *Try this* - A recommendation based on experience
- *Counterforce* - What causes you to stop applying the pattern
- *Examples* - Situations where the recommendation proved useful
- *Principles Involved* - Why the pattern appears to work
- *Related Up: Higher- and Down: Lower-level patterns*
- *Reading* - Further reading as referenced in the text.
- *Comments* - Comments from readers (like you) about the entry. Send your comments!