Patterns in a Nutshell

The "bare essentials" of Software Patterns

Brad Appleton *<bradapp@enteract.com>*

http://www.enteract.com/~bradapp/

1.0 What are Patterns?

Trendy:	Recent "hot topic", OOD buzzword, lots of hype!
Literary:	Form of software engineering problem-solving documentation
Pragmatic:	Describe practical solutions to "real world" problems
Recurring:	Identify good design structures which recur in practice
Generative:	Show how and when to apply the solution, and generate the desired design structure
Emergent:	Larger solutions emerge indirectly from applying patterns in succession, and in concert together

2.0 Pattern Origins and History

- Writings of architect Christopher Alexander
  (coined this use of the term *"pattern"* ca. 1977-1979)
- Documentation of best practices and handbooks for engineering and architecture
- Literate programming (Don Knuth), ca. 1984
- Kent Beck and Ward Cunningham, Tektronix, OOPSLA'87
  (used Alexander's *"pattern"* ideas for Smalltalk GUI design)
- Erich Gamma, Ph.D. thesis, 1988-1991
- James Coplien, Advanced C++ Idioms Book, 1989-1991
- Gamma, Helm, Johnson, Vlissides, (*"Gang of Four"*) Object-Oriented **Design Patterns** book, 1991-1994
- PLoP Conferences and books, 1994-present

3.0 Pattern Definitions

A "pattern" is ...

- An abstraction from a concrete form which keeps recurring in specific, non-arbitrary contexts. *[generic definition]*
- A recurring solution to a common problem in a given context and system of forces. *[Alexander]*
- A named "nugget" of instructive insight, conveying the essence of a proven solution to a recurring problem in a given context amidst competing concerns.
- A successfully recurring "best practice" that has proven itself in the "trenches".
- A literary format for capturing the wisdom and experience of expert designers, and communicating it to novices

4.0 Kinds of Software Patterns

- Design Patterns (software design; often object-oriented):
  - architecture (systems design)
  - design (component interactions)
  - programming idioms (language-specific techniques/style)
- Analysis Patterns (recurring & reusable analysis models)
- Organization Patterns (structure of organizations/projects)
- Process Patterns (software process design)
- Domain-Specific: *Any other domain you can think of!*

5.0 Pattern Elements

- **Name**
  - a meaningful "conceptual handle" for discussion
- **Context**
  - tells *how the problem occurs* / *when the solution works*
- **Problem**
  - statement of the problem / *intent* of the solution
- **Forces**
  - trade-offs, goals+constraints, motivating factors/concerns
  - tells *why the problem is difficult*
- **Solution**
  - tells *how to generate* the solution
  - the solution structure, its participants & collaborations

6.0 Pattern Elements (cont.)

- **Examples** (optional)
- **Resulting Context**
  - describes the end result, benefits and consequences
  - shows how the forces were balanced/traded-off
  - tells *how the solution works out*
- **Rationale** (optional)
  - underlying principles/heuristics justifying the solution
  - tells underpinnings of *why the solution works out*
- **Related Patterns**
  - patterns which are similar, or may precede/follow this one
- **Known Uses**
  - 3 or more independent instances of “real world” success

7.0 Why Patterns?

Software Patterns help us because they:

- Solve "real world" problems
- Capture domain expertise
- Document design decisions and rationale
- Reuse wisdom and experience of master practitioners
- Convey expert insight to novices
- Form a shared vocabulary for problem-solving discussion
- Show *more* than just the solution:
  - context (when and where)
  - forces (trade-off alternatives, misfits, goals+constraints)
  - resolution (how and why the solution balances the forces)

8.0 Summary - What Patterns Are *Not*

Software Patterns are ***not*** ...

- Restricted to software design or Object-Oriented design
- Untested ideas/theories or new inventions
- Solutions that have worked only once
- Any old thing written-up in pattern format
- Abstract principles or heuristics
- Universally applicable for all contexts
- A "silver bullet" or panacea

9.0 Summary - What Patterns *Are*

Software Patterns ***are*** ...

- *Recurring* solutions to common problems of design
- *Practical/concrete* solutions to real world problems
- *Context* specific
- "*Best-fits*" for the given set of concerns/trade-offs
- *"Old hat"* to seasoned professionals and domain experts
- A *literary form* for documenting best practices
- A *shared vocabulary* for problem-solving discussions
- An effective means of (re)using, sharing, and building upon *existing wisdom/experience/expertise*
- *Massively overhyped!*

10.0 Pattern Resources - Books

- **A Pattern Language: Towns, Buildings, Construction** (APL)
  Christopher Alexander; Oxford University Press, 1977
- **The Timeless Way of Building** (TTWoB)
  Christopher Alexander; Oxford University Press, 1979
- **Design Patterns: Elements of Reusable Object-Oriented Software** (GoF)
  Gamma, Helm, Johnson, Vlissides; Addison-Wesley, 1994
- **Pattern-Oriented Software Architecture: A System of Patterns** (POSA)
  Buschmann, Meunier, Rohnert, Sommerlad, Stal; Wiley and Sons, 1996
- **Pattern Languages of Program Design** (PLoPD1)
  Coplien and Schmidt (editors); Addison-Wesley, 1995
- **Patterns of Software: Tales from the Software Community**
  Richard Gabriel; Oxford University Press, 1996
- **Analysis Patterns: Reusable Object Models**
  Martin Fowler; Addison-Wesley, 1996
- **Pattern Languages of Program Design 2 (PLoPD2)**
  Vlissides, Coplien, and Kerth (editors); Addison-Wesley, 1996

11.0 Pattern Resources - Online

- Patterns Home Page, *http://www.hillside.net/patterns/*
- Patterns Discussion FAQ, *http://g.oswego.edu/dl/pd-FAQ/pd-FAQ.html*
- Ward Cunningham's WikiWikiWeb, *http://c2.com/cgi/wiki?WelcomeVisitors*
- Portland Pattern Repository, *http://www.c2.com/pp/*
- AGCS Patterns Page, *http://www.agcs.com/patterns/*
- Jim Coplien's OrganizationPatterns Front Page (a WikiWikiWeb clone), *http://www.www.bell-labs.com/cgi-user/OrgPatterns/OrgPatterns*
- Patterns Mailing Lists, *http://www.hillside.net/patterns/Lists.html*
- Cetus Links: Patterns, *http://www.objenv.com/cetus/oo_patterns.html*
- Brad's Pattern Links: *http://www.enteract.com/~bradapp/links/sw-pats.html*
- Brad's Patterns Intro: *http://www.enteract.com/~bradapp/docs/patterns-intro.html*
- Luke Hohmann's Patterns Intro: *http://members.aol.com/lhohmann/papers.htm*
- Doug Lea's OOD Patterns Intro: *http://gee.cs.oswego.edu/dl/ca/ca/ca.html*