

New Pattern Classification Approach

Ondrej Lehecka, Milos Kudelka, Vaclav Snasel
Dept. of Computer Science, VSB – Technical University of Ostrava,
17. listopadu 15, 708 33, Ostrava-Poruba
{Ondrej.Lehacka,Milos.Kudelka,Vaclav.Snasel}@vsb.cz

Abstract: In this paper we are focusing on patterns and pattern classification. We are dealing how patterns from different domain can relate in solving more complex tasks. We try to design pattern classification reflecting pattern features. We assume that it can be very helpful in pattern searching. We assign classification attributes to patterns and use them to find pattern clusters. Each pattern cluster should offer different solutions for the same problem. It is also possible to navigate in pattern space thru pattern clusters or to specialize pattern cluster towards problem being solved. Finally we introduce new community based pattern portal containing pattern references and implementation of presented approach.

What is pattern

In 70's Christopher Alexander formulated patterns for solving architectural tasks (see (1)). Pattern is not considered an idea how to solve the problem but it is considered as a description and generalization of the experience which leads to the method how to solve the problem (in the beginning there is practical experience not only an idea). The method must be so flexible so it can be used over and over again without stereotype results. Pattern is brief description of the problem and description of its general solution in particular context.

Alexander's definition of the pattern: *"Each pattern is rule describing relation between certain context, problem and solution."*

Pattern Domains

According to paper (2) we agree with general thesis that patterns need patterns need empiric proves for legitimacy of their use; patterns must be readable for their users; patterns can model many application domains; the use of patterns in software architecture, user interface and application domain of project can improve communication between interdisciplinary development teams.

We would like to demonstrate that working with patterns in wider context has sense. We are working with both well-known patterns in development community and less known patterns covering different fields (domains). We considered these pattern languages: **Design patterns** (3). They are applied in SW design and are focused to ensure program flexibility. **Analysis patterns** (4). These patterns are more or less independent on problem domain and describe solution of problems on conceptual level. **Patterns for enterprise solutions** (5) (6). They describe design problems of enterprise applications. **Integration patterns** (7) describes problems of integrating SW systems. **Unit Test Patterns** (8). Author is focusing on patterns for unit testing. **Software Configuration Management Patterns** (9) describe also versioning and other related tasks. **Patterns for design GUI** (10) (11). Important feature of well designed user interface is keeping ergonomic rules and standards which can be described using patterns. **E-learning patterns** (12). This catalogue of patterns is used for describing features which are expected from e-learning system and its content. **Pedagogical patterns** (13). During teaching student and teacher are often in the same situation. Their problems can be solved similarly using patterns. **Patterns for design of customer oriented websites** (14) are patterns useful for design and implementation of commercial web sites. **Pattern language for pattern writing**. Finally in our list of fields covered by patterns there is pattern language

where authors are focused on how to find patterns and how to correctly describe them, see (15).

Pattern classification

Gang of Four

Design patterns have been extended among people since 1995 when book (3) was published. The book is denoted by abbreviation *GoF* (Gang of Four). It is first well described and documented catalog of design patterns. It contains 23 patterns divided into three categories: *Creational patterns*, *Structural patterns* and *Behavioral patterns*.

Martin Fowler

In book (5) there are patterns for design and implementation of enterprise information systems. Author defines group of patterns covering different part of enterprise application design. In each group there are both patterns which can be used together for solution of certain problem and patterns which describe solution of certain problem in different way. Author divides patterns into eight groups: *Domain Logic Patterns*, *Data Source Architectural Patterns*, *Object Relational Patterns*, *Web Presentation Patterns*, *Distribution Patterns*, *Offline Concurrency Patterns*, *Session State Patterns* and *Base Patterns*.

Microsoft

In book (6) authors elaborate the patterns on different technological and implementation levels. They divide patterns into groups in three ways. First division is into *Clusters* (Web Presentation, Deployment, Distributed Systems, Performance and Reliability, Services). Second division is according to *Different Levels of Abstraction* (Architecture, Design, and Implementation). Third division is *Viewpoint* (Database, Application, Deployment, and Infrastructure). These three divisions exist concurrently and in their intersections there are *Frames* which contains group of patterns covering certain problem in detail. This approach gives the user much better information about where to pick up the pattern if he knows which problem he solves within whole system.

Internet Catalogs

MS Patterns & Practices

This pattern classification comes up from (16) considering patterns covering almost whole software developing process on Microsoft platform. One part of patterns called *implementation patterns* is describing implementation details and practical experiences of authors with the patterns on Microsoft .NET platform. Further the patterns are divided according to the abstraction levels. Based on this division authors created the *Enterprise Architectural Space Organizing Table*. Important is that pattern organization was created in contribution with community members and their opinions.

www.patternshare.org

Authors of this website are trying to help developers by organizing patterns into tables according to the known classifications and authors. Beside this it is possible to insert new patterns and participate in discussions. This portal is focusing only on patterns for software solutions and allows categorize patterns using few predefined categories.

Classification Problems

Basic feature of current classifications is that each one is focused and developed only within one domain. If we want to observe patterns across different domains current classifications are not sufficient. Each author of a pattern book is trying to divide set of patterns into few

classes. These classes or categories are dependent on author's point of view or on field which the patterns cover.

Studying patterns needs different classification. Especially in situation when we do not certainly know in which context we should search. Each pattern should contain besides *descriptive part* which is for study purposes also a *classification part*. From *descriptive part* the reader will learn the pattern name, relating problem, context, solution, consequences, etc. *Classification part* contains information which categories does the pattern belong to. Both parts of pattern description are important. It is clear that the descriptive part is almost fixed. While classification part can be subject of discussion and can change according to the different perspectives.

Consequence: Pattern classification in each book is useful for studying and for orientation in the problem domain.

Sometime you need a solution of problem which you did not meet before. You can solve the problem from the scratch but it is better to find appropriate pattern. This approach minimizes possible failure. If you don't know the pattern then the existing available classifications might not be very helpful for searching the pattern. For example let's imagine situation when we are searching for pattern for enhancing computation independence on used algorithm and we don't know design patterns (specifically pattern *Strategy*). Information that the pattern belongs to group of behavioral patterns is not very useful.

Consequence: Existing pattern classifications don't help in searching of unknown pattern for a given problem.

After studying patterns for longer time some relations between different patterns from different domains appear. There are some new publications which collect patterns from different domains for solving some larger, more complex tasks (see (17)).

Consequence: None of the pattern classifications gives the overview about how patterns from different areas relate to each other.

Our contribution

The key problem for us is the process of finding existing pattern. Our goal is to supply techniques and tools to be able to find patterns as effectively as possible with knowledge of the problem being solved. We are dealing with following tasks:

Flexible classification

The classification should be reflecting features which the pattern brings. The classification must be opened and flexible enough to be able to adapt new perspectives. The pattern classification is based on set of classification attributes and their assignment to patterns. These attributes are evident after longer studying of patterns and subject of professional discussion. Most valuable attributes are those spawning many domains. Examples of classification attributes can be: according to what pattern addresses – *solves efficiency | flexibility | coupling | robustness*; according to required degree of knowledge – *for beginner | intermediate | expert*.

Web portal

According to presented approach the web application has been built. It is community based pattern web catalog with support for pattern classification and intelligent searching. Support for pattern classification is built on ability of creating classification attributes and possibility

of assigning attributes to patterns. The application allows searching of patterns both using common database and full-text searching and using sophisticated filtering of classification attributes. Users can influence pattern classification.

Navigation in pattern space

For orientation in pattern catalog it is important to have tool for creating of pattern groups and be able do see relations between groups. To solve this problem it seems that clustering analysis methods are appropriate. For this purpose we selected Formal Concept Analysis, see (17). For FCA the input is matrix of patterns and their classification attributes. Each computed concept contains set of patterns and set of attributes. The concept can be interpreted as a set of patterns with the same features which are described by classification attributes.

Navigation structure is created with concepts and edges connecting them. From properties of FCA it is implied that on such structure we can move from any cluster to any cluster using two directions – “up” and “down”. If we will move using “down” direction then we will reach cluster containing more attributes than the starting cluster. We can consider the “down” direction as *concretization direction* called *specialization* – (see figure 1) because we get cluster defined by bigger set of attributes than the previous cluster. It means that it contains more criteria which mean smaller set of patterns. In navigation using “up” direction the reverse mechanism apply.

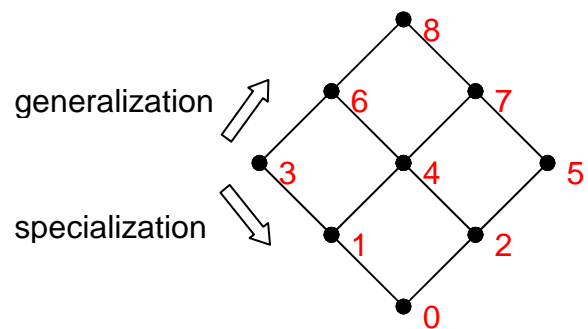


Figure 1: Types of navigation

Experiments

The portal started as a diploma work [see (18)] and is being still developed. We studied tens of patterns and consult their categorization with domain specialist. According to the information we get we created groups of attributes. There are 14 groups at the moment in our system and it is highly probable that by enriching system with other patterns and domains there will be need to extend the set of attributes.

Group	Attribute
Pattern addresses efficiency	object/source recycling concurrent data access Data access strategy support scalability data exchange method
Pattern addresses flexibility	algorithm independence HW/SW platform independence operation independence reuse extending behavior
Pattern addresses coupling	object-relation mapping inter-object communication object collaboration coupling of application parts
Roles	architect designer developer tester
Level of knowledge	expert intermediate beginner

Table 1: Example of classification attributes

Architectural pattern	GUI pattern
Pattern: Model-View-Controller	Pattern: Overview Plus Detail
Source: Enterprise Patterns (Fowler)	Source: Designing Interfaces (Tidwell)
Attributes:	Attributes:
Suitable for building user interface	Suitable for building user interface
Suitable for web applications	Suitable for web applications
Suitable for enterprise solutions	Organizes content

Table 2: Example of pattern classification

We inserted more than 500 patterns from mentioned sources and 140 classification attributes.

There is need for permanent revision of attributes and need for inventing new attributes. In this respect the implemented system brings one key feature which is the ability of propose, review and accept new attributes.

Conclusion

Our aim is to present pattern portal and its features to professional community. It is available on address www.pattron.net. We try to insert a lot of patterns, classify them with known classifications. Our effort was to design attributes and create environment for experiment. Our portal allows users and communities to contribute in finding useful classifications.

References

1. **Alexander, Ch.** *A Pattern Language: Towns, Buildings, Construction*. New York : Oxford University Press, 1977.
2. *Interaction Design Patterns: Twelve Theses*. **Borcher, J.**
3. **Gamma, E., Helm, R., Johnson, R., Vlissides, J.** *Design Patterns – Elements of Reusable Object-Oriented Software*. s.l. : Addison-Wesley, 1995. ISBN 0-201-63361-2.
4. **Fowler, M.** *Analysis Patterns. Reusable Object Models*. s.l. : Addison-Wesley, 1997. ISBN 0-201-89542-0.
5. —. *Patterns of Enterprise Application Architecture*. s.l. : Addison-Wesley, 2003. ISBN 0-321-12742-0.
6. **Trowbridge, D., Mancini, D., Quick, D., Hohpe, G., Newkirk, J., Lavigne, D.** *Enterprise Solution Patterns using Microsoft.NET Version 1.0*. s.l. : Microsoft Corporation, 2003.
7. **Hohpe, G., Woolf, B.** *Enterprise Integration Patterns*. s.l. : Addison-Wesley , 2003. ISBN 0321200683.
8. **Clifton, M.** Advanced Unit Test, Part V - Unit Test Patterns. *Code Project*. [Online] 2004. <http://www.codeproject.com/gen/design/autp5.asp>.
9. **Berczuk, S., P., Appleton, B.** *Software Configuration Management Patterns: Effective Teamwork, Practical Integration*. s.l. : Addison-Wesley, 2002. ISBN 0-201-74117-2.
10. *Vzory pro HCI a GUI*. **M., Kudělka.** Ostrava : s.n., 2004, Vol. Sborník konference Tvorba softwaru 2004. ISBN 80-85988-96-8.
11. **Tidwell, J.** UI Patterns and Techniques. [Online] <http://time-tripper.com/uipatterns/>.
12. E-learning patterns. [Online] http://www2.tisip.no/E-LEN/patterns_info.php.
13. Pedagogical patterns. [Online] <http://www.pedagogicalpatterns.org/>.
14. **Van Duyne D. K., Landay J. A., Hong J. I.** *The Design of Sites: Patterns, Principles, and Processes for Crafting a Customer-Centered Web Experience*. s.l. : Pearson Education, 2002. ISBN 020172149X..
15. **Meszaros, G., Doble, J.** A Pattern Language for Pattern Writing. [Online] <http://hillside.net/patterns/writing/patternwritingpaper.htm>.
16. Describing the Enterprise Architectural Space. [Online] <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnpag/html/entarch.asp>.
17. **Van Duyne D. K., Landay J. A., Hong J. I.** *The Design of Sites: Patterns, Principles, and Processes for Crafting a Customer-Centered Web Experience*. s.l. : Pearson Education, 2002.
18. **Ganter, B., Wille, R.** *Formal Concept Analysis: Mathematical Foundations*. s.l. : Springer, 1999.
19. **Lehecka, O.** Support for searching software patterns. Olomouc, Czech Republic : Palacki University, 2006.