International Journal of Engineering and Technical Research (IJETR)
ISSN: 2321-0869, Volume-2, Issue-4, April 2014

Unified Design Pattern Template

S.S.Suresh, Dr.Kamewswara Rao, Prof.Dr.V.K.Saraswat

***Abstract*— Design pattern selection is important task for implementing quality software. Patterns are grouped into several categories. Finding a suitable pattern from a category is a challenging task. Each category of patterns is described in a template form. Template conveys purpose and essentials of design pattern. Without exploring a template, pattern selection would not be appropriate. There are near about 10 templates proposed in the literature. The current paper describes those 10 templates, and their limitations. Subsequently proposes a Unified Pattern Template (UPT), which is a enhanced version of previously defined templates.**

***Index Terms*— template form**

I. INTRODUCTION

A pattern is a common solution to a recurring problem in a software design [1]. Software developers need to know about design patterns for software development. Design patterns evolves from software design principles (e.g. Coupling, Cohesion etc.) [9]. Christopher Alexander laid foundation for design patterns [1] [2]. His work inspires GoF (E.Gamma, R.Helm, Ralpha Johnson and J.Vlissides) for developing design patterns. GoF describes 23 design patterns and represents them in UML notations (Unified Modeling Language) for object oriented systems. These 23 patterns have become popular and widely used. Due to its necessity, some of the Universities in India and abroad introduces design pattern course in masters and degree programs. In this paper, the word "design pattern" and "pattern" represents the same. According to GoF design pattern is a "Descriptions of communicating objects and classes that are customized to solve a general design problem in a particular context". They are variety of patterns, for instance:

Algorithm strategy patterns: These patterns address high-level strategies describing how to exploit application characteristics on a computing platform [14].

Computational design patterns: These patterns address concepts related to key computation identification [14].

Execution patterns: These patterns address concepts related to supporting application execution, including strategies in executing streams of tasks and building blocks to support task synchronization [14].

Manuscript received April 20, 2014.
S.Suresh , Presently working with I2IT Pvt Ltd promoted by Finolex Industries, Pune.
Dr.Kameswara, associate professor, Electronics and Computer Engineering, K.L University, Vijayawada
Prof. Dr.V.K. Sarawat, asst. Professor in Dr. B.R.Ambetkar University, Department of Computer Science, Agra, India

GoF patterns fall into execution patterns. In addition to the above mentioned pattern types, domain specific patterns are evolving rapidly. For instance, user interface design patterns describes problem and solutions related to user interface design. It is limited to user interface only. Similarly, database patterns describes database accessing and data security patterns. In the fall 1994, GoF developed a pattern catalog containing three categories namely Creational, Structural and Behavioral [12]. The catalog creation was part of E.Gamma Ph.D thesis [12]. Pattern description mainly contains three parts (PCS): problem statement, context and solution ('P' represents problem statement, 'C' represents context and 'S' represents solution). Often pattern details get documented using Pattern Form (also called pattern template). A PatternForm is a format and structure used to write a pattern [15]. There is variety of template forms available. Based on the granularity (level of detail), they can be divided as: Short-template forms and long-template forms. Short-term template forms are simple, easy to understand but difficult to apply. Whereas long-term template forms are lengthy but conveys adequate information. The details of common template forms is given in the section II

Problem statement describes a statement about the problem that a pattern solves. Context describes a situation in which the problem occurs. Solution describes solution or technique for solving the problem. Every design pattern description contains PCS. User of a pattern must understand the logical connection between P, C, and S. Evaluating problem statement (P) is important task for pattern selection". Truly, template is as important as implementation. Because template gives detailed description about a patterns including pros and cons of a design pattern. There are 10 most widely used templates, namely Alexandrian form, Pattern catalog form, POSA form, Portland form, Coplin formand P of
FF form etc. The researcher consolidates these templates and creates a new template called Uniform Pattern Template (UPT) for searching design patterns in a information retrieval system . Design patterns descriptions are available in various formats such as PDF, HTML, printed text books etc. Generally, text books give in detail description and are useful to beginners. In real-time, a designer or architect require instantaneous solution to design problem. To provide instantaneous solution, it is proposed an Information Retrieval System based on the concepts of Decision Support System (DSS) [6] [15]. For a quick view, an overview of the Information Retrieval System - technical architecture is given in the following Fig. 1.1

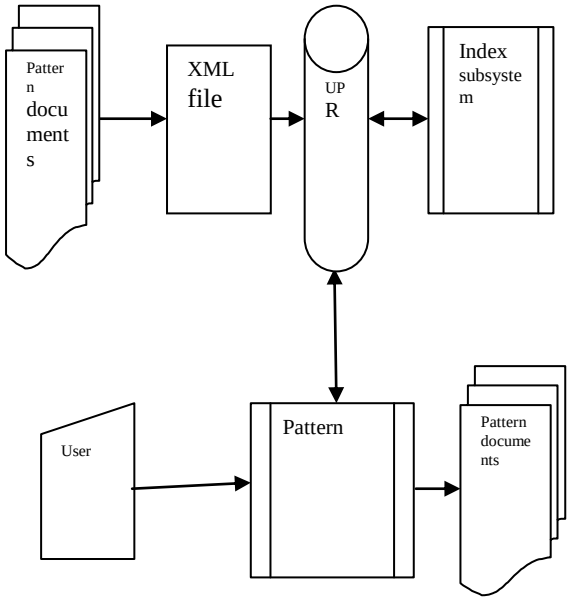

Fig: 1.1 Technical Architecture

Often, Pattern designer documents pattern details using a template. Pattern designer selects a template and chooses granularity for describing patterns. The fig 1.1 depicts the architecture adopted for creating a repository of design patterns for storing in information retrieval system for automatic pattern retrieval. The researcher intention in this paper is to give details about UPT and its creation process and does not describe index subsystem, pattern selection subsystem, and user input. The rest of the paper has been organized as follows:

Section II gives literature review of pattern template forms, section III gives UPT creation process, section IV gives conclusion.

II. Literature Review

There are at least 10 template forms for design pattern description [4] [15] namely Alexandrian Form, Canonical form, Coplien form, GoF form,Compact form,CockburnPm form,Portland form, BeckForm,FowlerForm and CoreJ2EEForm. These template forms evolves over the period of time. Each template has structure, granularity, advantage and disadvantage. There is no standardization for template forms. It is open concept. Template forms have commonalities with one another. Various authors proposed template forms with their own assumptions. Ultimately, template format selection remains as a choice. Pattern writing and pattern selection are not the same. However, a person can be pattern writer and pattern selector but a pattern selector may not be a pattern writer. Pattern writer's view differs from pattern selector/user view. The current paper emphasis pattern selector/user view. The following table I shows the common template forms and granularities.

Table I pattern forms

Pattern form	Elements of Description
Alexandrian	Title, Prologue, Problem statement, Discussion, Solution, Diagram, and Epilogue.
GoF	Intent, Motivation, Applicability, Structure, Participants, Collaborations Consequences, Implementation, Sample code, Known Uses, and Related Patterns
POSA	Summary, Example, Context, Problem, Solution, Structure, Dynamics, Implementation, and Example resolved, Variants, known uses, and Consequences.
Portland	Problem, Context, and forces.
Coplien Form	Problem, Context, Forces, and Solution.
P of EAA form	How it works, when to use it, and one or more examples.
Compact	Context, Problem , Forces, Solution, and Resulting Context
Beck form	Title, context, problem, forces, solution, and resulting
Flower Form	Title, summary, and discussion
Canonical form	Name, Alias (optional), Problem, Context , Forces, Solution, Example (optional?), Resulting Context , Rationale (optional) , Known Uses And Related Patterns

The table I includes the Alexandrian form, Pattern Catalog form, POSA [Pattern Oriented Software Architecture] form, Portland form, Coplin form, and P of FF form. The purpose of each template form is as follows:

In 1997 Alexandrian proposed a form called Alexandrian form [14]. The purpose of the Alexandrian form is to guide users to generate solutions for the problems. It is the first template form which has become basis for all other template forms. It has limited structure. It does not describe consequences of applying a pattern. If consequences are not known, then applying a pattern will be risky. Similarly, every pattern must balance forces. A force describes constraint that should be balanced while implementing a pattern. Omitting forces leads to implementation failures

In 1998 Eric Gamma et al proposed a form called GoF form [8]. It is popular template form. The purpose of this form is to help users to create solutions to problems but less focused on when to apply a pattern [15]. This template form mainly focuses on dynamics of a pattern. It has long structure. This template gives in detailed explanation of a pattern, including implementation examples. Hence, it conveys pattern knowledge to wide number of users.

In 1996 Pattern Oriented Software Architecture proposed a form called POSA form [4] [10]. This form is structured and

lengthy. It does not focus on dynamics like applicability, forces, programe code, and structure of a pattern. Hence, it does not cover wide number of users.

In 1987 Howard G. (Ward) Cunningham proposed a form called Portland form. This form is narrative and short. It does not focus on pattern applicability, consequences, structure, and implementation; through they are important for pattern selection. Portland format authors submitted a research paper on Portland template form to the Pattern Languages of Programs conference (PLoP) [15]. This template form emphasis on forces and solutions for patterns. The advantage of Portland form is, it collects and connects patterns so that they will be studied and understood as a whole [15].

In cope proposed a form called CoplienForm. It is also called Canonical form. Cope was one of the popular pattern writers. He prefers to write patterns in a short description. It focuses less on dynamics like applicability, consequences, when to use a pattern, structure, and implementation [14].

In 2006 Martin Flowler et al proposed a form called P of EAA (Patterns of Enterprise Application Architecture). This form very short and brief. It describes when to use a pattern, how to use a pattern and an example. This form given abstract level details. It does not describe issues like collaborations, applicability's, related patterns etc. Novice users find difficulty in following this. It focuses on enterprise application patterns. These patterns concerned with the issues: enterprise application layer, domain logic, logic to a relational database, web based presentation, and principles in distributed design.
Compact form describes patterns in a simple and short form. It limits pattern description to one-page. It mainly focuses on the context of pattern and resulting context after applying a pattern. Resulting context is analogous to consequences. The resulting context is important to understand the state of a pattern. Usually, resulting context leads to find new patterns [15].

In 2007, KentBeck proposed a form called CompactForm. It is compact, means; a pattern description is set limited to one-page. Hence, it has elements, Context, Problem, Forces, Solution, and Resulting Context. It conveys adequate information regarding a pattern but missing implementation and consequences. Generally, designer or architect perceives consequences after applying a pattern.

Fowler proposed a form called FowlerForm. It is also very short form like other forms (compactForm, BeckForm, and CopeForm). It contains elements: title, summary and discussion. It conveys inadequate information for applying a pattern. It has smallest structure compared to all other structures. It may be easy to read but difficult to apply.

The researcher has done detailed study of the above said templates and found that finding a suitable pattern for a given problem statement is complex task. Because, granularity of templates differ from each other. However, pattern selection is important than implementation and successful implementation requires programming skills. Keeping this point in view, the researcher simplifies the existing pattern formats and writes a new template format containing attributes of pattern selection and is called Unified Template Format (UTF). The UTF simplification involves addition of new elements, deletion of unwanted elements, and simplifying common elements.

The above stated templates have common elements. The common elements are: problem and solution. Excluding common elements, all other elements in a template do not play equal role. Only a few are important for pattern selection and the rest are useful for implementation. Software architects give importance to the design issues rather implementation. Because, incorrect design leads to incorrect implementation. So, design pattern selection is important as it is a design issue. Hence, the researcher, of this paper has done detailed study about pattern selection criteria to identify the key elements responsible for the selection and proposed a Unified Pattern Template. It is revised work of our earlier publication [15]. It is a fairly direct emulation of GoF form with some simplification. The next section explains the procedure for the creation of UP

III. UPT CREATION

The UPT creation process is described as follows: -

a) List Elements of each the standard pattern template
b) Identify synonymous elements: In some templates, the elements have been named differently though the purpose is same. For example, Motivation is synonymous to Force. So, synonymous elements have been renamed to a common element. The table II shows the list of synonymous elements identified across the 10 template described in the section b. The table 3.1 shows existing element on the left side and its synonymous element on the right side. To make the process of identification of required elements simpler, existing elements are replaced with its respective synonymous elements.

Table II Synonymous elements

Existing element	Synonymous element
Problem	Problem statement
Title	Name
Diagram	Structure
Motivation	Force
Context	Applicability
when to use	Applicability
Example	Known uses
Solution	Implementation
Examples	Known uses
Examples resolved	Known uses
how it works	Implementation
Resulting context	consequences

c) Based on the objective of the system being developed, few of the elements have not been considered i.e Deleted.

These are: Implementation, Prologue, and Epilogue, Collaborations (or collaborators), Participants, Sample code, Discussion, Structure, Summary, and Variants.

The above stated elements are suitable for implementation. Hence, they are not considered for creating the UPT. In other way, they are deleted from keeping in UPT. The element "implementation" refers "pattern implementation". It expresses implementation techniques. The element "sample code" gives "example implementation" using a programming language. "Discussion" is general statement. Authors (pattern writers) write their opinions in the form of discussion. The element "Structure" describes classes and their relationship. It is useful to develop the application quickly. The element "Summary" is synonymous to "conclusion". The element "Epilogue" represents beginning of pattern whereas the element "prologue" represents ending of a pattern. Epilogue and Prologue are boundaries of a pattern. The element "Collaboration" is an implementation issue. It gives the list of objects which are collaborating each other to perform a particular task. The element "Participants" give participating objects. It is synonymous to collaborators.
d) Simplified elements in the UPT are:

Name, Intent, problem statement, force, applicability, known uses, relevant pattern, consequences.

Apart from these standard elements, some researcher defined elements have been added to the UPT to develop search driven effective Information Retrieval System.

These selected elements are result of unification of all the elements in different standard template formats described above. Thus, it is named as Unified Pattern Template (UPT) and the repository which stores the patterns organized in this UPT form is called Unified Pattern Repository (UPR). The details of UPR have been given in earlier publication [15]. The UPR is an amalgamation of standard elements and researcher defined elements. Researcher defined elements are: definite keyword, generic question, structural question, answer type, and functional domain. Researcher defined elements enhances the scope of the pattern search and strengthens pattern selection. The following sections describe Meta data of UPT.

A) UPT Elements

The UPT contains two types of elements namely standard elements and researcher defined defined elements. Standard elements are: name of the pattern, type of a pattern, intent, problem statement, forces, applicability, consequence, Author, related pattern, known uses,variation of the pattem,version number of the patten. Meta data of each of standerd element is given in the table III.

Table III Metadata of standard elements

Element	**Description**
Name	It helps to refer a pattern
Intent	It signifies the goal behind the pattern
Problem statement	It signifies the situation in which a pattern can be applied
Forces(motivation)	It is a scenario consisting of a problem and a context in which this pattern can be used.
Applicability	It signifies situations in which a pattern is usable: the context of the pattern.
Consequences	It signifies tradeoffs and side effects of pattern caused by using a pattern.
Known use	It signifies examples of real usages of the pattern
Author	
Related Pattern	It signifies other patterns that have relationship with the pattern.

The researcher defined elements includes definite keyword, structural questions and generic questions. The UPT does not focus on implementation details of a pattern. Because, the goal of this UPT is to provide suitable pattern to the user problem or scenario when problem statement is given in text format. In order to provide improved search, additional elements like name of the pattern, type of the pattern, author and version number are added to UPR. The following table IV shows Meta data details of the researcher defined elements.

Table IV UPT Elements

Definite Keyword	Keyword signifies most important words through which the search system retrieves pattern documents quickly without performing text parsing.
Generic Question	Generic Questions are the questions related to a pattern. The questions reflect charectertics of a pattern. Questions can be single choice, multiple choices; fill in the blank and true/false type. A question requires answer and a question may have dependant question. Each question is given alphanumeric code (Ex: CQ01).
Structural question	Structural question indicates a question related to a structure of a pattern. The structural question emphasis on structure of a pattern. It improves pattern identification and strength the pattern selection.
Answer type	There are two types of answer type included. There are Boolean or Text.
Functional Domain	It explains the list of functional domains in which patterns are already used.

Type	The type signifies the category in which a pattern belongs (e.g. algorithm design pattern, execution pattern, computational pattern etc).
Author	It signifies name of the author who developed the pattern.
Pattern category	The category to which pattern belongs. This would help users to choose pattern correctly. For example, structural pattern describe structure of a pattern related to a design concept within the subsystem. Whereas architectural pattern describes subsystems and there interconnectivity (Ex: MVA architecture).
complexity	It describes difficulties in implementing a design pattern. It is analogous to forces.
When pattern fails	It describes conditions that lead to failure of a pattern.

B) UPT Limitations

As it is stated above in section 3, UPT is an amalgamation of standard elements, unified from existed templates formats and researcher defined elements. The UPT is best suited for search driven systems.

IV. Conclusion

Some of the elements (definite keyword/keywords/generic questions etc) in the UPT are good enough when search is based on keywords. Other elements are useful in general. In future, we would try to improve the existing template. In the literature sections, for some of the template forms, Year of introduction is not given. In case, if any reader come across those details, Pls mail me sssuresh74@gmail.com.

ACKNOWLEDGMENT

I thank Himalayan University, Department of Computer Science, and Arunachal Pradesh, India for giving opportunity to purse research work (doctoral program) in there campus.

REFERENCES

[1] Albit-Amiot, H.P.Cointe, Y.G. Guehenuue, and N.Jussien, 2001, November. Instantiating and detecting design patterns: Putting bits and pieces together. In 6 Th annual International Conference on Automated software Engineering (ASE), 166-173.

[2] A. Sengupta, S. Mohan, R. Doshi. XER - Extensible Entity Relationship Modeling. In Proceedings of the XML 2003 Conference, p. 140-154. Philadelphia, USA, December 2003.

[3] Bernadette Farias Lóscio, Ana Carolina Salgado, Luciano do Rêgo Galvão. "Conceptual Modelling of XML Schemas", Proceedings of the fifth ACM international Work shop on Web information and data management (WIDM03), 102-105, ACM Press, 2003.

[4] Dirk Reihle and Heinz Zullighoven, "Understanding and using patterns in Software Development", Theory and practices of Object systems, 2, 1996.

[5] G. Psaila. ERX: A Conceptual Model for XML Documents. In Proceedings of the 2000ACM Symposium on Applied Computing, p. 898-903. Como, Italy, March 2000.

[6] S.S.Suresh, Dr.M.M.Naidu," Design Pattern Recommendation system, (Methodology, Data model, Algorithms) Proceedings of International Conference on Computational Technique and Artificial Intelligence (ICCTAI'2011), Planetary Scientific Research Center, Oct 7-8 Pattaya, Prentice Hall, 1998.ISBNN0 Thailand.

[7] L. Rising, "The pattern Almanac 2000", Addition Wesley publications, 2000, .ISBNNO: 0201379678.

[8] Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Summerland and Michael Stal. Pattern-Oriented Software Architecture: A System of Patterns, Volume1, Wiley publications, 2007.

[9] Scott Brain,"Net Objectives Design Pattern repository", [online]available:http://www.netobjectivestest.com/PatternRepository /index.php?title=Main_Page,last modified 27 June 2012.

[10] Cunningham & Cunningham,"Port land Pattern epository", [online] available:http://c2.com/ppr last accessed 19 12 2012.

[11] Michael Angeles KONIGI, "Design pattern repositories", [online] available at http://konigi.com/wiki/design- pattern- repositories, last edited 17 November 2011.

[12] "The Hillside Group, [online] available : http://hillside.net.

[13] S.S.Suresh, Dr.sagar Jambhorkar," Unified Pattern Repository (UPR) for Design pattern Selection, International Journal of Engineering Research and Development**, e**-ISSN: 2278-067X, **p**-ISSN: 2278-800X, Volume 5, Issue (January 2013), PP. 01-08. Pattern forms, available: http://c2.com/cgi/wiki?PatternForms, last edited April 2005.

[14] Software Design pattern, available: http://en.wikipedia.org/wiki/Software_design_pattern, last updated 30 March 2014.

[15] Pattern Languages of Programming, Conference proceedings (annual, 1994)

S.S.Suresh: **He has** 14+ years of experience in academics (MCA/M.TECH/MBA). He has published articles, research papers in Journals, conferences proceedings and received foreign University invitations to deliver lectures'. He has done MCA from Nagarjuna University and M.Tech (CS) from JRN University, Rajasthan. Presently working as Ass.Professor, ASCT department, International Institute of Information Technology (I^2IT), Pune, promoted by Finolex Industries, Pune.

Dr.Kameswara Rao is working as associate professor, Electronics and Computer Engineering, K.L University, Vijayawada; He has completed his Ph.D from Budhle Khand University in computer science. Proior to this, he has completed MCA and M.Phil in computer science.

Dr.V.K. Prof. Saraswat is working as asst. Professor in Dr. B.R.Ambetkar MCA and PHD in computer science from Agra University. He has vast experience in teaching and research. His research areas are: Software Engineering, and Computer networks.