PARADE

A pattern-based knowledge repository for parametric designs

SHENG-FEN CHIEN¹, HSIU-PAI SU² and YU-WEI HUANG³
¹,²,³ National Cheng Kung University, Tainan, Taiwan
{schien, n76024058, pa6024031}@mail.ncku.edu.tw

Abstract. It is important to formalize design knowledge to capture tacit design experience and techniques. This research aims to utilize the power of patterns and language to formulate knowledge of parametric design. We have found through our own experience of learning and teaching parametric design, examples are the most familiar form of learning. We proposed a way of documenting design knowledge in four parts: pattern, example, case and source. We have implemented the repository as a web browser based system, named PARADE. A preliminary study of the system is conducted.

Keywords. Design pattern; knowledge repository; parametric design.

1. Introduction

It is important to formalize design knowledge to capture tacit design experience and techniques. “A Pattern Language” by Alexander and colleagues (1977) establishes a system to formulate and organize design knowledge through patterns. The book can be considered the first pattern-based knowledge repository. In a book form, the relationships between patterns can only be presented through names and indexes of pattern. The software design patterns book by Gamma et al. (1995) constitutes another repository of software design knowledge. Again in a book form, some reusable software design example code can only be shown on book pages as printed texts or downloaded separately. Woodbury and his research team created a website documenting design patterns for parametric modelling (Woodbury, 2010b). There, the form of patterns is enriched with step-by-step illustrations/animations to present procedural knowledge. In addition, the relationships between patterns...
can be associated through hyperlinks. The use of digital media makes the patterns more accessible to designers.

We hope to utilize the power of patterns and pattern languages to formulate knowledge of parametric design. Furthermore, the knowledge should be documented in such a way that can be easily accessed and disseminated to designers. Our research objectives are: first, to identify a set of well-defined format to record design patterns and pattern languages; second, to collect a comprehensive list of parametric design patterns; and third, to create a pattern management tool for manipulating patterns, navigating through pattern libraries, and adding relationships between patterns to create pattern languages.

2. Pattern Form

The concept of design patterns originated in architecture (Alexander et al, 1977). In recent decade, the concept has been adapted to software engineering (Gamma et al, 1995), systems engineering (Pfister et al, 2011), interaction design (Borchers, 2001, Tidwell, 2005), web usability (Graham, 2002), education (Baggetun et al, 2004), communication (Schuler, 2008), and so on. Popular pattern forms evolved from Alexander et al (1977), which was adapted by Gamma et al. (1995) for software design and Tidwell (2005) for HCI, to Woodbury’s patterns for parametric modelling (Qian et al, 2007, Woodbury, 2010b, Woodbury, 2010a). Although there are other pattern forms (refer to Kruschitz and Hitz, 2009 for additional examples), we consider the above ones to have largest impact in the parametric design patterns. Woodbury Form is directly applicable to our research. In particular, it demonstrated uses of animated visuals to illustrate the application of patterns.

In terms of documenting the relationship between patterns so as to maintain “pattern languages”, Woodbury Form has only the “Related Patterns” to record the simple association relationship between patterns. Alexander Form is under the “A Pattern Language” scheme to record the relationship of several types: within town, building, construction scale, or cross town-building, building-construction scale. Gamma Form qualifies the relationship with annotations (Figure 1).

Kruschitz and Hitz (2009) conducted an extensive review on HCI design patterns and argued that, in addition to standardizing the pattern form to ensure the content quality of design patterns, relationships between patterns are essential part to best profit from the full potential of design patterns. In addition, they stressed the need for pattern management tools to support authoring and manipulation of pattern libraries. Khwaja and Alshayeb (2013) went further to propose a design pattern definition language for software engineer-
ing patterns. They used Gamma Form as the foundation and added the information of pattern owner, author, pattern version, and language. These four types of information are important when considering tools for pattern management.

Figure 1. The overall structure of software design patterns (taken from (Zimmer, 1995))

3. Related Work

Woodbury and Qian (Woodbury, 2010a, Qian, 2009, Qian et al, 2008, Qian et al, 2007) had explored design patterns for parametric modelling. In particular, (Qian, 2009) methodically went through and tackled issues of knowledge acquisition, pattern authoring, communication through patterns, and evaluation of patterns. The research was conducted with a small group of designers and the effects are hard to find as the time past. Furthermore, the “pattern language” aspect was not explored in their work.

The power of design pattern methodology lies in the formation of pattern languages. A critical point of forming pattern languages is the relationships between patterns (Zimmer, 1995, Noble, 1998). Kruschitz and Hitz (2009), Schobert and Schümmer (2006), and Schuemmer (2003) illustrated various ways of visualizing the relational structure (or network) within a pattern lan-
language to assist designers navigating in the network and search for relevant patterns. This will be the key step to turn parametric design pattern collection into pattern language.

Jessop (2004), Derntl and Motschnig-Pitrik (2005) argued for pattern language as a framework for learning. Schuler (2008), and Riehle (2003, 2011) tried in their practices and examined the benefit of using patterns in design communications. As a relatively new design method in the field of architecture, parametric design pattern language may be quite feasible.

4. Pattern, Example, Case and Knowledge Source

A knowledge repository for parametric design is a knowledge management tool. Therefore, our system needs to provide the functionality of design pattern management. We decided to start with the Woodbury Form. From our previous experience of learning and teaching parametric design using “Elements of Parametric Design” (Woodbury, 2010a), we found that samples are the most familiar form of learning. Taking these considerations altogether, we proposed a way of documenting design knowledge in four parts: pattern, example, case, and knowledge source (Figure 2). A pattern, as defined by Alexander and Woodbury, describes a solution to a problem in an abstract form. An example (“sample” in Woodbury Form) gives concrete illustrations of an instance of a problem with a particular solution. A case is a design project that can demonstrate composite use of patterns. A knowledge source (or simply, source) records the sources or references that lead to the creation of a pattern, an example, or a case.

A pattern is formulated like the pattern in Woodbury Form. It consists of a name to describe the intent, an image to represent the pattern, an intent section to state the objective, a usedWhen section to describe a problem and a context, a why section to describe reasons to use, and a how section to provide the step-by-step procedure of solving the problem. In addition, a pattern may have relatedPatterns that reference other patterns, knownUses that link to examples, and references that refer to sources.

Examples and cases are formulated like the sample in Woodbury Form with an additional part to record the computer language/platform on which the example or case is implemented. An example consists of a name, an image, an associatedPattern, a purpose section to explain the intent, a platform section, an implementation section to store the code/program, and a references section. A case has all the parts of an example, except the associatedPattern is replaced by associatedPatterns, as well as a relatedExamples section is added. A source is formulated using the BibTeX format.
In terms of their relationship, a pattern can be demonstrated through one or more examples. Each example illustrates the use of one corresponding pattern. A case illustrates the composite use of two or more patterns, each of which in turn links to examples. Each example or case is associated with one or more sources.

5. An Evolving Knowledge Repository

We acquired the knowledge of parametric design in architecture from two sources: literature and designers. Our very first sets of design patterns were those published by Woodbury and his research team (Woodbury, 2010a, Qian et al, 2008). A minor set of patterns was selected from Lai (2011). In addition, we invited designers to discuss their designs and reviewed the design process they had gone through. We formulated design patterns according to the data collected and reviewed the pattern with designers iteratively to create new patterns.

We collected books on parametric design or computational design related themes. The content was analyzed first to identify examples and cases of existing patterns. Each candidate example is recreated using a predetermined platform (in this case Rhino+Grasshopper), annotated with descriptions,
checked against the how section of target pattern, and revised. The revision process involves at least two researchers and three iterations before committing the case into the repository. Each candidate case has gone through a similar process to confirm its addition into the repository.

We expect the repository to evolve continuously with new examples, cases, and patterns. The repository currently contains 18 patterns, 43 examples, and 3 cases. The knowledge is extracted from 13 distinct sources. All examples and cases are demonstrated using Rhino+Grasshopper. Figure 3 illustrates a snapshot of the knowledge base (patterns in black ovals, examples in rectangles with light edges, cases in rectangles with bold texts, sources in grey circles) at the time of writing.

6. The PARADE System

We have implemented the repository as a web browser based system, named PARADE (i.e. PARAmetric DEsign). We employ HCI design patterns (Tidwell, 1997) to guide and examine the interaction and interface designs. The interaction design of PARADE is based on responsive website design so that we may provide the same user experience on both desktop and mobile devices. This conforms to the HCI: Liquid Layout pattern (Figure 4). The home page (Figure 5) is laid out with tiles of examples (HCI: Grid of Equals). Additional information is displayed dynamically as users scroll down the

---

Figure 3. A snapshot of the knowledge base in the repository.
window pan (HCI:Infinite List). Users may browse patterns, examples, and cases (HCI:Module Tabs), as well as add them as personal favorite.

Figure 4. PARADE on phone, pad, and laptop displays conform with HCI:Liquid Layout.

Figure 5. PARADE home page design employs HCI:Module Tab, Grid of Equals, Infinite List.

The pattern page displays the image, name, intent / use when / why, how, related pattern/examples/cases, and source respectively from top to bottom. The example page and the case page are laid out similarly: the image, name, pattern(s) used, purpose, and how to do in slideshow-style display. Currently, all examples and cases are demonstrated using Rhino+Grasshopper. In our personal experience, as well as from observations of others, parametric designers constantly switch focuses between model view and program view,
indicating these two views are equally important to users. With the constraints of display real estate, the slideshow is arranged so that users may toggle between model view and program view (HCI: Alternative View). The step (how to do) section on example and case pages needs to convey algorithms. Therefore, the step-by-step illustration is necessary. To maintain user’s focus and context, we employ HCI: Sequence Map pattern to achieve the desire user interaction (Figure 6).

The backend service supports the pattern management module and the user account management module. The pattern management module employs a two-tier authoring process: any power user may create new pattern/example/case/source, but to publish them requires an administrative staff to approve. For literature-type sources, the system can parse bibliography entries to avoid mistakes from manual inputs.

7. Discussions and Conclusion

We have conducted a preliminary study on the PARADE website to see how students may use it without clear learning instructions. Two groups of students were given building façade modelling problems. One group has a simple problem that requires uses of 3 patterns to solve, and the other group has a difficult problem that requires at least 5 patterns to solve. A small number of students (about 30%) were able to find examples comparable to their problem and tried to learn from examples. It is clear that we need to setup instruction materials on how to use the system.

Furthermore, students mentioned the need to see “the whole picture” of how patterns, examples and cases are related to each other. We consider this issue is related to the formation of pattern language. In the step section of cases, we have created diagrams that illustrate the usage connection between
patterns (Figure 7). Such a diagram help users to establish “a picture” of the algorithm used when solving a complex problem. However, we do not have such diagrams for the whole repository.

Figure 7. An overview diagram of patterns used in a case.

In summary, we have accomplished our objectives on developing a pattern management tool and collected a fair amount of parametric design patterns. However, we have not established a meaningful relationship structure between patterns to form a pattern language. We are continuously acquiring related design knowledge from practicing architects and designers to identify new patterns, as well as collecting more examples and cases. Leaned from the related work, we need to provide visualization and navigation support for relationship structure in our pattern language.

Acknowledgements
The research is in part supported by MOST Taiwan (SC 102-2221-E-006 -224 -MY2).

References


