Y. Ikeda, C. M. Herr, D. Holzer, S. Kaijima, M. J. Kim. M, A, Schnabel (eds.), *Emerging Experience in Past,Present and Future of Digital Architecture, Proceedings of the 20th International Conference of the Association for Computer-Aided Architectural Design Research in Asia CAADRIA 2015,* 000–000.

AN EMPIRICAL FOUNDATION FOR DESIGN PATTERNS IN PARAMETRIC DESIGN

RONGRONG YU[1] and JOHN S. GERO[2]
[1,] *University of Newcastle, Newcastle, Australia*
rongrong.yu@newcastle.edu.au
[2,] *University of North Carolina at Charlotte, Charlotte, USA*
john@johngero.com

Abstract. This paper presents the results from exploring the impact of using a parametric design tool on designers' behaviour in terms of using design patterns in the early conceptual development stage. It is based on an empirical cognitive study in which eight architectural designers were asked to complete two architectural design tasks with similar complexity respectively in a parametric design environment (PDE) and a Geometric modelling environment (GME). Protocol analysis was employed to study the designers' behaviour. To explore the development of design patterns during the design process, we utilise the technique of Markov model analysis. Through Markov models analysis of the PDE and GME results, we found that there are significantly more Function to Structure transitions in PDE than in GME. During this transition process, designers select an existing structure/solution for the particular function/design problem based on their experience or knowledge, which is a process of applying an existing design pattern to the problem. From this result we can infer that when architects apply programming and scripting in their design, such as in a PDE, they exhibit the characteristic of using design patterns.

Keywords. Design pattern; parametric modelling; protocol studies.

1. Introduction

Parametric design has become increasingly prevalent in architectural design. However, there is a lack of empirical evidence supporting an understanding of designers' behaviour in parametric design environments (PDEs). In a PDE designers often adopt existed design patterns based on their experience of using their design knowledge and the experience of using parametric tools. This phenomenon has not been adequately studied and evaluated in architecture design. To improve our understanding of the possible use of design patterns while designing, a cognitive study in which designers were asked to complete two architectural design tasks with similar complexity respectively in a PDE and a GME is presented. Protocol analysis (Ericsson and Simon 1993, Gero and Mc Neill 1998) was employed to study the designers' behaviour. From the Markov model analysis of the resulting protocol, the results of how designers use design patterns in the PDE are presented and discussed.

2. Background

2.1 PARAMETRIC DESIGN

Parametric design is a dynamic, rule-based process controlled by variations and parameters, in which multiple design solutions can be developed in parallel. According to Woodbury (2010), it supports the creation, management and organization of complex digital design models. By changing the parameters of an object, particular instances can be altered or created from a potentially infinite range of possibilities (Kolarevic 2003). The term "parameters" means factors which determine a series of variations. In architecture, parameters are usually defined as building parameters or environmental factors. In the architectural design industry, parametric design tools are utilized mainly on complex building form generation, multiple design solution optimization, as well as structural and sustainability control. Currently, typical parametric design software includes Generative component from Bentley Corporation, Digital project from Gehry Technology, Grasshopper from McNeel. Scripting tools include Processing based on the Java language, Rhino script and Python script, based on VB language from McNeel. In this study, Grasshopper was chosen as the parametric design environment. Grasshopper is both an advanced environment for facilitating conceptual design and it is in relative wide-spread use in the architectural profession.

2.2. FBS ONTOLOGY

As one of the main design ontologies, Gero's FBS model (Gero 1990) has been applied in many cognitive studies (Gero and Tang 1999, Jiang 2012, Kan 2008, Kan and Gero 2009, Pauwels et al 2015). Researchers argue that it is potentially capable of capturing most of the meaningful design processes (Kan and Gero 2009) and the transitions between design issues are clearly classified into eight design processes. The FBS ontology (shown in Figure 1) contains three classes of variables: Function (F), Behaviour (B) and Structure (S). Function (F) represents the design intentions or purposes; behaviour (B) represents the object derived (Bs) or expected from the structure (Be); and structure (S) represents the components that make up an artefact and their relationships. The model is strengthened by two external design issues: requirements (R) and descriptions (D). The first of these represents requirements from outside design and the second, descriptions, mean the documentation of the design. Figure 1 shows the FBS ontology indicating the eight design processes—formulation, analysis, evaluation, synthesis, and three reformulation I, II, III. Among the eight design processes, the three types of reformulation processes are suggested to be the dominant processes that potentially capture innovative or creative aspects of designing by introducing new variables or new directions (Kan and Gero 2008). The FBS ontology is claimed to be a universal coding scheme for various design environments (Kan and Gero 2009). By calculating the transitions between design issues, various analyses can be conducted. In this study, the FBS ontology was used as the basis model of analysis for developing the coding scheme in the protocol study that was used to capture and produce behavioural data.

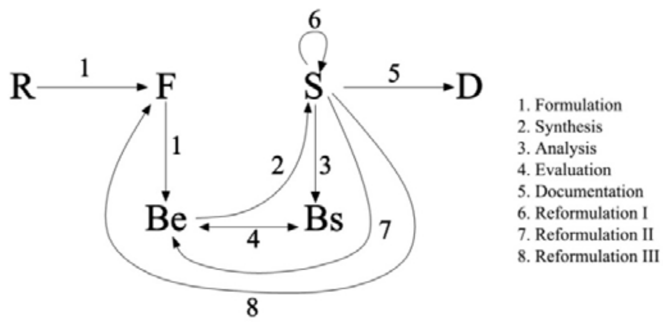


Figure 1. The FBS ontology (Gero and Kannengiesser 2004).

3. Experiment setting

Eight designers participated in the experiment, each of whom is a professional architect with an average eight years of experience in architectural design, and no less than 2 years on parametric design.

During the experiment, both designers' activities and their verbalizations were video-recorded. There were two design sessions: one session used Rhino as the Geometric Modeling Environment (GME) and the other session used Rhino and Grasshopper as the Parametric Design Enivironment (PDE). Designers were given 40 minutes for each design session. Task 1 is a community centre design and Task 2 is a shopping centre design, both containing specific function requirements. A pre-modelled site was provided to the designers, Figure 2. Because the present study wass focused on exploring designers' behaviour at the conceptual design stage, designers were required to only consider concept generation, simple site planning and general function zoning. No detailed plan layout was required. The design sessions and tasks were randomly matched among different designers. During the experiment, designers were not allowed to sketch manually so that almost all their actions happened on the computer to ensure that the design environment is purely within the PDE and GME. This allows us to minimize the impacts of other variables and focus on the two different design environments for comparison during the comparative analyses.

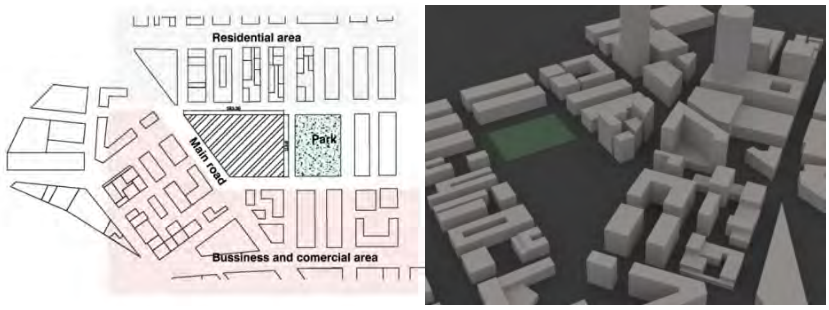


Figure 2. Site model provided to the designers during the experiments

4. Analysis Results

4.1. GENERAL RESULTS

This protocol study employed an integrated segmentation and coding method. The segmentation and encoding process are based on the "one segment one code" principle (Pourmohamadi and Gero 2011). It means there is no overlapped code or multiple codes for one segment. If there are multiple codes for one segment, the segment will be further divided. Table 1 provides the general information of the coding coverage. The numbers shown in the table are the average of the eight protocols. The average overall segments are respectively 244 in the PDE and 224 in the GME. Designers also spend more time in the PDE session (48 min) than in the GME (44 min). On average over 92.2% of segments can be coded as FBS codes. Segments not coded included those associated with communication and software management.

The protocol analysis takes the video and verbal utterances of the participants and transforms them into a sequence of semantic symbols, called design issues, based on the FBS coding scheme. This sequence of design issues becomes the basis for all quantitative results that follow.

Table 1. General coding information.

	Design environment	Time (mins)	Number of Segments	Coded Percentage (%)	Speed (Segments/min)
Mean	GME	44	224	92.24	5.11
	PDE	48	244	92.20	4.78
SD	GME	11.22	45.32	4.29	1.20
	PDE	7.43	29.71	3.54	0.53

4.2. 1ST ORDER MARKOV ANALYSIS RESULTS

The Markov model describes the probabilities of moving from one state to another (Ching and Ng 2006, Meyn and Tweedie 2009), it demonstrates the tendency of future design moves. Kan and Gero adopt the Markov chain model using the FBS ontology to describe cognitive design processes (Kan and Gero 2009, Kan and Gero 2010). Within the context of the FBS ontology, the Markov matrix is a quantitative tool to study design activities based on the transition probabilities between design issues. It can also be used to study transitions between design processes. There are two types of Markov models of interest here: the 1st order Markov model and the 2nd order Markov model. The 1st order Markov model expresses the probability of moving to a future state depending only on the current state, without considering the past state, Figure 3. The 2nd order Markov model expresses the probability of moving to a future state depending on the current state and the previous state. This paper only presents the results of the 1st order Markov model.

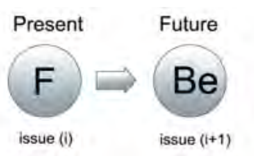


Figure 3. An example of the 1st order Markov model using the FBS ontology

The values in the Markov model are calculated using *LINKODER,* a software tool (LINKODER 2011, Pourmohamadi and Gero 2011). The matrices of the 1st order Markov model in the GME and the PDE are presented in Table 2. The numbers in the table are the average values of transition probabilities of the eight designers.

Table 2. The 1st order Markov model analysis

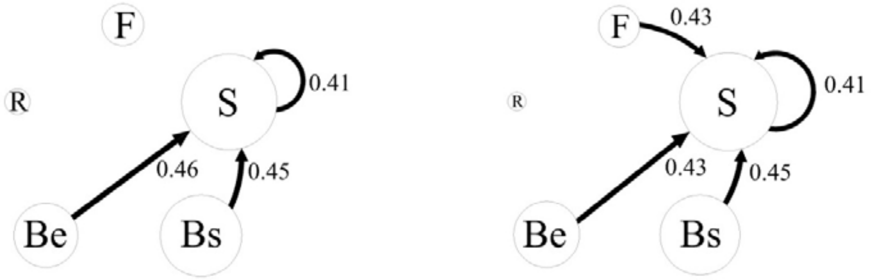
	R		F		Be		Bs		S	
	GME	PDE	GME	PDE	GME	PDE	GME	PDE	GME	PDE
R	0.20	0.20	0.18	0.30	0.39	0.21	0.11	0.03	0.11	0.26
F	0.03	0.02	0.23	0.11	0.29	0.26	0.23	0.19	0.23	0.43
Be	0.03	0.01	0.10	0.09	0.18	0.17	0.24	0.31	0.46	0.43
Bs	0.02	0.01	0.06	0.07	0.22	0.22	0.25	0.26	0.45	0.45
S	0.01	0.01	0.05	0.06	0.14	0.19	0.40	0.34	0.41	0.41

5. Design patterns in the parametric design environment

5.1. F TO S DESIGN MOVE

A descriptive diagram of the 1st order Markov model analysis in the GME and the PDE is given in Figure 4.. The circles labelled with the FBS codes represent the design issues, and the size of a circle represents the frequency of occurrence of that design issue. Each arrow shows the transition from one state to the other, and the thickness of the line represents a measure of the transition probability between design issues. To demonstrate the main activities of the designers, we select the transitions with the probability value larger than 0.4 and show them in Figure 4. The probability of 0.4 is selected as threshold to abstract the model, this threshold is 2 times of random probability. In the FBS model, each variable has 4 other states to go to, which means that the random probability is 0.2. The graphs in Figure 4 shows that the main patterns of design moves are Be-S, Bs-S, and S-S; the transition probabilities are very similar between the GME and the PDE. The primary difference between the GME and the PDE is that the transition probability from F to S is above the threshold in the PDE and below the threshold in the GME.

Within the context of the FBS ontology, this process of transitioning directly from function (F) to structure (S) is excluded from routine ways of design (excluded from the eight design processes expressed in FBS model). Previous research suggests from the study of software designers' behaviour, F to S is a typical design process that occurs frequently. During the F – S process, designers select an existing structure/solution for the particular function/design problem based on their experience or knowledge, which is the process of applying a learned, existing design pattern to the problem.



(a) (b)

Figure 4. (a) Transitions above the threshold in the 1st order Markov model in the GME, (b) main transitions of the 1st order Markov model in the PDE

Since software designers use design patterns when programming and scripting (Gamma, Helm et al. 2002, Fowler 2003), we can infer that when architects apply programming and scripting in their design, such as in a PDE, they would exhibit the similar characteristic of using design patterns. Design patterns are an important concept in both architectural design and software design. In software design, it assists software designers in working more efficiently and makes programming and scripting process traceable. In the PDE, if we can generalise these transitions to design patterns it would be of assistance to architects in their scripting process.

5.2. DESIGN PATTERNS IN THE PARAMETRIC DESIGN ENVIRONMENT

From Markov model analysis results, we found that design patterns are adopted more frequently in the parametric design environments than geometric modelling environment. The idea of design patterns was first introduced by Christopher Alexander: "each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice." (Alexander, Ishikawa et al. 1977, p x). That is, a pattern is a documentation of a solution suitable for certain kind of design problems which may occur frequently.

Patterns usually comes from designers' experience (Fowler 2003), which can be seen as a "induction" process. Designers generalise examples from their own design experience or from observations of other designers, and abstract the problem-solution pair, and formalise these into "patterns" which could be re-used. Those generated patterns could be improved, combined into a network of connections depending on design purpose (Alexander 1979). Woodbury, Aish et al. define that: "A pattern is a generic solution to a well-described problem. It includes both problem and solution, as well as other contextual information." (Woodbury 2010, p 185). A design expert has accumulated a large number of examples of problems and solutions in a specific domain (Razzouk and Shute 2012). The pattern itself is abstract, when designers apply the patterns, designers would revised the patterns to their own preference, or to the specific context of designing.

In the software design domain, educators found that Alexander's work on design patterns provided a strong foundation for re-usable software design. For instance, Gamma et al. (2002) defines patterns as a tool to describe compositional ideas in computer programming. This matches our analysis results that in parametric design more design patterns were used than in geometric design alone. Therefore we can infer that a feature of programming in a parametric design might affect the utility of design patterns during computational design process.

6. Conclusion

This paper has presented a protocol study which explores the phenomenon of using design patterns in a parametric design environment (PDE) and geometry modelling environment (GME). From the study we found that there is more design patterns used in the PDE than in the GME. Since the main difference between the two design environments is that rule algorithms are used in the PDE and not in the GME, we can assume that the rule algorithm feature affects the development and use of design patterns during the design process. That is to say, in a rule algorithm related design environment, designers tend to adopt the existing patterns based on their experience.

Acknowledgements

This research has been supported in part by the National Science Foundation grant CMMI-1161715. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of National Science Foundation.

Reference

Alexander, C.: 1979, *The Timeless Way of Building*, Oxford University Press.

Alexander, C., et al.: 1977, *A pattern language : towns, buildings, construction*. New York, Oxford University Press.

Ching, W. K. and M. K. Ng: 2006, *Markov chains: Models, algorithms and applications.* New York, Springer.

Ericsson, K. A. and H. A. Simon: 1993, *Protocol analysis : verbal reports as data*. Mass., MIT Press.

Fowler, M.: 2003, *Patterns of Enterprise Application Architecture*, Addison-Wesley.

Gamma, E., et al.: 2002. Design patterns: abstraction and reuse of object-oriented design in B. Manfred and D. Ernst (eds). *Software pioneers*, Springer-Verlag New York, Inc.**:** 701-717.

Gero, J. and U. Kannengiesser: 2004, The situated function-behaviour-structure framework. *Design Studies* 25(4), 373-391.

Gero, J. and H.-H. Tang: 1999, Concurrent and Retrospective Protocols and Computer-Aided Architectural Design. *CAADRIA1999*, Shanghai.

Gero, J. S.: 1990, Design prototypes: a knowledge representation schema for design. *AI Magazine* 11(4), 26–36.

Gero, J. S. and T. Mc Neill: 1998, An approach to the analysis of design protocols. *Design Studies* 19(1), 21-61.

Jiang, H (2012) *Understanding senior design students' product conceptual design activities,* PhD Thesis, National University of Singapore, Singapore.

Kan, W. T,: 2008, *Quantitative methods for studying design protocol*s, PhD Thesis, The University of Sydney, Sydney.

Kan, J. W. T. and J. S. Gero: 2008, Acquiring information from linkography in protocol studies of designing. *Design Studies* 29(4), 315-337.

Kan, J. W. T. and J. S. Gero: 2009. Using the FBS ontology to capture semantic design information in design protocol studies in J. McDonnell and P. Lloyd (eds). *About: Designing. Analysing Design Meetings*. New York, Taylor & Francis**:** 213-229.

Kan, J. W. T. and J. S. Gero: 2010, Exploring quantitative methods to study design behavior in collaborative virtual workspaces. *Proceedings of CAADRIA 2010*.

Kolarevic, B.: 2003, *Architecture in the digital age : design and manufacturing*. New York, NY, Spon Press.

LINKODER: 2011, url: http://www.linkoder.com/ (formerly called LINKographer)

Meyn, S. P. and R. L. Tweedie: 2009, *Markov chains and stochastic stability*. Cambridge, New York: Cambridge University Press.

Pauwels, P., Strobe, T., Derbroven, J. and De Meyer, R (2015) Conversation and critique within the architectural design process: A linkograph analysis, in J. S. Gero and S. Hanna (eds), *Design Computing and Cognition'14,* Springer, pp. 141-160.

Pourmohamadi, M. and J. S. Gero: 2011. LINKOgrapher: An analysis tool to study design protocols based on FBS coding scheme in S. Culley, B. Hicks, T. McAloone, T. Howard and Y. Reich (eds). *Design Theory and Methodology*. Glasgow, Design Society**:** 2-294-303.

Razzouk, R. and V. Shute: 2012, What Is Design Thinking and Why Is It Important? *Review of Educational Research* 82(3), 330-348.

Woodbury, R.: 2010, *Elements of Parametric Design*. New York, Routledge.