Studying Software Design Cognition

Jeff Wai Tak Kan
Krasnow Institute for Advanced Study
Fairfax, USA
Taylor's University, Malaysia
Email: kan.jeff@gmail.com

John Gero
Krasnow Institute for Advanced Study
Fairfax, USA
Email: john@johngero.com

***Abstract*—This paper presents the results of utilizing a generic coding scheme in the protocol analysis of a professional software design team. The paper commences by introducing the notion that the fundamental processes of designing are the same irrespective of the domain and of the artifact being designed. The Function-Behavior-Structure (FBS) ontology is proposed as the basis of a generic protocol analysis coding scheme. The rich set of results obtained indicates the utility of using an ontologically-based coding scheme in studying software designers. The results from this case study are compared with those from another case study involving designers from the domain of architecture.**

Keywords: Design, cognition, protocol analysis

I. Introduction

Over the past three decades, protocol analysis has become one of the most widely used methods to study human design activities and cognitive design processes [1] [2]. However, unlike many other research domains there is a lack of agreement in both the terminology and the research methodology. In studying designers there is no uniformity in the segmentation and coding of protocols. The papers in DTRS2 [1] and DTRS7 [2] demonstrate the range of methods and coding schemes applied to studying the same raw data. This paper uses a generic coding scheme to study a team of professional software designers. By using this generic method, the results from this study can be used to compare with other studies that use the same generic scheme.

II. FBS Design Ontology

In order to establish a common ground to study design activities, we propose to use an ontology as an overarching principle to guide the protocol study. Gruber [3] defines ontology, apart from its use in philosophy, as an explicit specification of a conceptualization. Knowledge of a domain is represented in a declarative formalism in terms of a set of objects and their relationships. An ontology may be thought of as the framework for the knowledge in a field.

A design ontology is described by defining the representational terms and their relationships. Gero [4] viewed designing as a purposeful act with the goal to improve the human condition. Gero [4] stated: "The meta-goal of design is to transform requirements, more generally termed functions which embody the expectations of the purposes of the resulting artefact, into design descriptions. The result of the activity of designing is a design description."

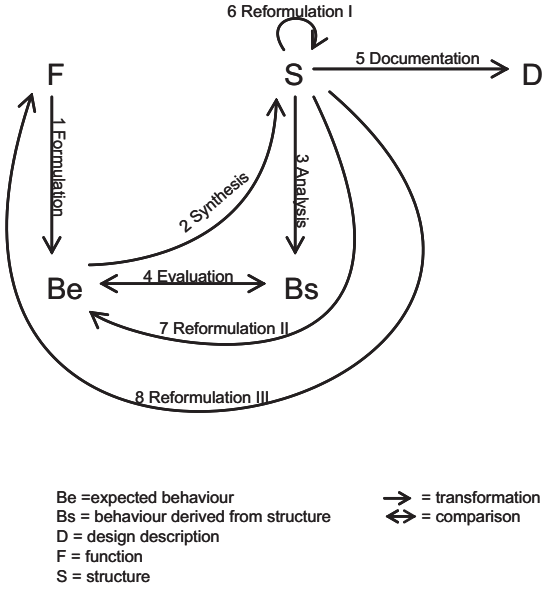

Fig. 1. The FBS ontology of design processes

The coding of the protocols will be based on a general design ontology, namely, the Function-Behavior-Structure (FBS) ontology [4] as a principled coding scheme (as opposed to an ad hoc one). Figure 1.

The FBS design ontology [4], as a formal model, represents designing in terms of three fundamental, non-overlapping, classes of concepts called design issues: function or purpose (F), behavior (B), and structure (S); along with two external classes: design descriptions (D) and requirements (R). In this view the goal of designing is to transform a set of functions into a set of design descriptions. The function of a designed object is defined as its teleology; the behavior of that object is either expected (Be) or derived (Bs) from the structure (S) that is the components of an object and their relationships. A design description cannot be transformed directly from the functions, which undergo a series of processes among the FBS variables. Figure 1 shows the eight FBS design processes in relation to the function (F), behavior (B), and structure (S) state variables. Formulation (F→Be) is the transformation of the design intentions, expressed in term of functions, into

expected behavior. Synthesis (Be→S), is the transformation of the expected behavior (Be) into a solution structure. Analysis (S→Bs) is the derivation of "actual" behavior form the synthesized structure (S). Evaluation (Bs↔Be), is the comparison of the actual behavior (Bs) with the expected behavior (Be) to decide whether the solution structure (S) is to be accepted. Documentation (S→D), is the production of the design description.

Traditional models of designing iterate the analysis - synthesis - evaluation processes until a satisfactory design is produced. The aim of introducing the three types of reformulations is to expand the design state space so as to capture the innovative and creative aspect of designing. These have not been well articulated in most models because they have not been adequately understood.

Reformulation type I (S→S'), addresses changes in the design state space in terms of structure variables or ranges of values for them. Reformulation type II (S→Be'), addresses changes in design state space in terms of behavior variables. A review of synthesized structure may lead to the addition of expected behavior variables. Reformulation type III (S→F'), addresses changes in design state space in terms of function variables.

A. Software Designing and FBS

The basis of the FBS view is that all design tasks, irrespective of domain, can be described using a generic framework such as the FBS framework as any design task will have functions or purposes (F) to fulfill for which the object is being designed, and resulting behaviors (Be and Bs) and structure (S). Therefore, one significant advantage such an approach has over those not based on an ontology is that the FBS approach can be used to represent designing from various domains under a common ontological structure.

In designing physical objects, the manifestation of structure (S) variables will usually be some physical aspect. For example, to design a portable shelter, the function (F) of shelter can be formulated to an expected behaviors (Be) of a space with simple erection method that provides protection. This may be synthesized into an "A-frame" structure (S). With this structure, one can analyze its behavior (Bs), for example headroom, and structural stability. After evaluating these behaviors the designer may accept or reject this structure (S).

However, in the design of software the structure (S) will not have any physicality. In a typical object-oriented programming paradigm, the objects or patterns of objects and their relationships are the structure. The software designers formulate the expected behaviors of objects from the functions of resulting program. With these expected behaviors they can synthesize the objects or the relationships of codes of those objects. With these objects they can derive their behaviors by either running that part of the program, formally or mentally simulating their behaviors. For example in this protocol which is the result of designing a traffic signal simulator (based on the design requirements outlined in http://www.ics.uci.edu/design-workshop/files/UCI_Design_Workshop_Prompt.pdf), the input, roads and intersections are objects that were considered as part of the structure (S). During designing there were discussions concerning the expected behavior, such as: drag and drop environment of the input, and signal timing of the intersections. An example of behavior derived from structure is the time for cars to travel between intersections.

It is useful to think of the concepts of $independent$ and $dependent$ variables in a software design formalism to provide an analogy to the relationships between structure (S) and behaviors (Be and Bs). An independent variable is one that may be varied freely or autonomously by a designer. Using the simple example of a formula that computes area ($area = length \times breadth$), $length$ and $breadth$ are examples of the independent variables. A dependent variable is one that derives from the interaction that occurs between the independent variables, such that its behavior derives from the changes that are effected through the independent variables. $Area$, therefore, is a dependent behavior that derives from the individual changes that a designer makes to $length$ or $breadth$. In our coding of the protocol of software designers, structure (S) comprises those variables that the designers manipulate independently (or conceive that the users of the program may wish to do). In this protocol, examples include number of lanes, cars and intersection geometry. Behavior comprises those properties that derive from these, for example, the number of cars that can clear through the intersection in a given amount of time.

III. Observations of the Session

In this section we present some basic qualitative and quantitative observations of the protocol session labeled "anonymous" in the NSF Workshop on Studying Professional Software Design (http://www.ics.uci.edu/design-workshop/index.html). Since we are concerned with showing the applcability and utility of using a generic method to study software design cognition the specifics of the design task are not of particular interest and are not repeated here. They can be found at http://www.ics.uci.edu/design-workshop/files/UCI_Design_Workshop_Prompt.pdf.
Similarly, the protocol videos and the transcripts of the utterances in the protocol videos are not presented here. They can be found at http://www.ics.uci.edu/design-workshop/videos.html.

A. Qualitative

We refer to the two participants as Male 1 and Male 2. Male 1 seemed to be more senior and took the leadership role; he started by asking Male 2 to give a summary and how he saw the program structure could be broken down. Male 1 did most of, if not all, the drawing and writing on the whiteboard. They spent more than 5 minutes at the beginning to protocol the design brief individually without verbalizing.

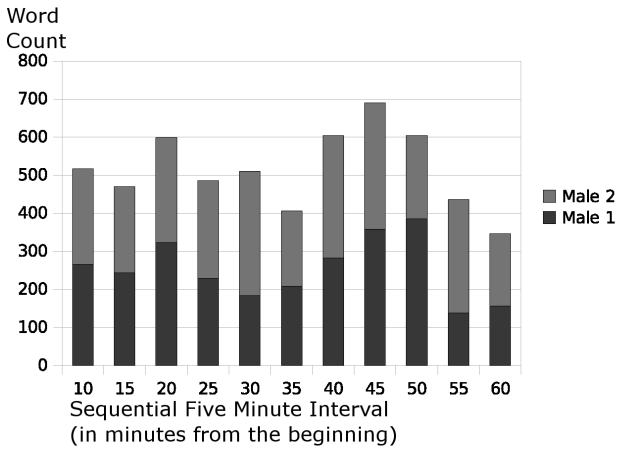

Fig. 2. Word count in sequential five minute intervals over the protocol for Male 1 and Male 2

B. Quantitative

The turn taking is very even, Male 1 had 125 utterances and Male 2 had 123 utterances. The word counts of Male 1 and Male 2 are 2777 and 2894 respectively. Figure 2 shows the word count of the two individuals in 5 minute intervals without breaking up the utterances.

IV. FBS Design Issues Segmenting and Coding Scheme

The generic coding scheme applied here consists of function (F), expected behavior (Be), behavior derived from structure (Bs), structure (S), documentation (D) and requirement (R) codes. The protocols are segmented strictly according to these six categories. Those utterances that do not fall into these categories are coded as others (O); these include jokes, social communication, and management.

This framework does not assume any domain of designing nor does it assume any particular number of participants being studied. It only abstracts and describes the state of affairs of designing in terms of FBS design issues of a segment. With this generic coding scheme, although simple, we aim to capture some fundamental aspects of designing which will form a foundation for further analysis. Unlike other coding systems, there is a nexus between segments and codes using this coding scheme: one segment per code and one code per segment. This forms the basis of segmentation. This is likely to produce a higher degree of uniformity and robustness than other methods since the same design issues are always used.

The segmentation/coding involved two independent coders who segmented/coded separately (in different geographical locations with no inter-coder bias or influence) and who then carried out an arbitration session to produce the final arbitrated segmented/coded protocol. The agreement between the two coders was 82.8% and between the coders and the final arbitrated results was 90.4% respectively. The high percentages of agreement provide additional support for the claim that the FBS ontology based coding methodology provides a logically coherent way to view a protocol by different people. In absence of such a unifying guiding ontological framework, different coders can end up producing widely differing segmentation schemes making any kind of quantitative analysis not robust and, more significantly, results from individually generated coding schemes cannot be compared unless the codes can be directly mapped on to each other.

V. Results and Analysis

A. Descriptive Statistics

The segmented protocol contains 640 segments, 603 of these are design issue codes, those that do not relate to design issues are coded as "O" and not counted further in this analysis. Figure 3 shows the distribution of the FBS codes. Male 2 refers to the requirements more than Male 1 but overall the "R" coded segments were sparse (17 in total). There were only three function (F) issues being raised (Male 1: 2, Male 2: 1). Male 1 did most of the documentation, where by "documentation" we mean the externalization of thoughts by drawings, symbols and words.

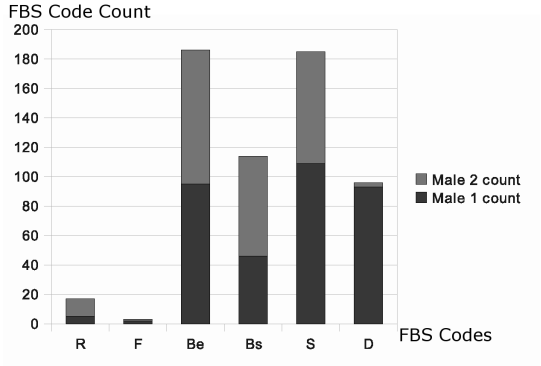

Fig. 3. The distribution of FBS codes for Male 1 and Male 2

The design session was divided into three equal sized thirds based on the number of segments to provide the basis for a more detailed analysis. Figure 4 shows the change of distributions of the FBS codes across the three thirds. Changes can be observed qualitatively in this figure and statistical analyses to confirm the qualitative results can be carried out.

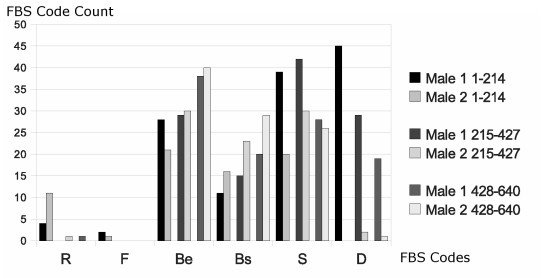

Fig. 4. The FBS code distributions for the beginning, middle and end of the session for Male 1 and Male

B. Sequential Analysis

Traditionally design protocols have been analyzed by static statistical methods that often are based on the assumption that each segment is an independent event. Markov chains analyze or describe the probability of one event leading to another. Kan [5] proposed using Markov analysis to examine the sequence of FBS events. Kan and Gero [6] used Markov analysis to compare three design sessions.

Markov chains had also been used to analyze writer's manuscripts and to generate dummy text [7]; for the ranking of web pages by Google [8]; and to capture music compositions and synthesize scores based on the analyzes [9], to name some of its applications.

A probability matrix P is used to describe the transition of the six FBS states (R, F, Be, Bs, S and D). Equation 1 shows the probability matrix of this software design session.

$$P = \left(\begin{array}{c|cccccc} & R & F & Be & Bs & S & D \\ \hline R & 0.29 & 0.00 & 0.12 & 0.00 & 0.41 & 0.18 \\ F & 0.00 & 0.00 & 0.67 & 0.00 & 0.00 & 0.33 \\ Be & 0.00 & 0.01 & 0.21 & 0.23 & 0.39 & 0.16 \\ Bs & 0.02 & 0.00 & 0.47 & 0.15 & 0.29 & 0.07 \\ S & 0.03 & 0.01 & 0.26 & 0.20 & 0.19 & 0.31 \\ D & 0.01 & 0.00 & 0.42 & 0.15 & 0.43 & 0.01 \end{array}\right) \quad (1)$$

There are only 3 occurrences of the F event causing a roundoff to zero in many of the rows and columns containing F. The occurrences of R are also low with some roundoff to zero as well. P is one characterization of the design session. It shows that the most likely transition for this design team is from thinking about the purpose of the design (F) to its expected behavior Be.

Dividing the design session into three equal sized thirds based on the number of segments allows us to compare the design behaviors during the beginning, middle and end of the session to determine if there are changes in behavior as the session progresses. P_1, P_2 and P_3 are the probability matrices of the segments from 1 to 214, 215 to 427 and 428 to 640 respectively for the three thirds of the session.

$$P_1 = \left(\begin{array}{c|cccccc} & R & F & Be & Bs & S & D \\ \hline R & 0.33 & 0.00 & 0.13 & 0.00 & 0.33 & 0.20 \\ F & 0.00 & 0.00 & 0.67 & 0.00 & 0.00 & 0.33 \\ Be & 0.00 & 0.02 & 0.09 & 0.16 & 0.42 & 0.30 \\ Bs & 0.04 & 0.00 & 0.40 & 0.16 & 0.28 & 0.12 \\ S & 0.07 & 0.04 & 0.16 & 0.14 & 0.16 & 0.44 \\ D & 0.03 & 0.00 & 0.40 & 0.18 & 0.43 & 0.00 \end{array}\right) \quad (2)$$

$$P_2 = \left(\begin{array}{c|cccccc} & R & F & Be & Bs & S & D \\ \hline R & 0.00 & 0.00 & 0.00 & 0.00 & 1.00 & 0.00 \\ F & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ Be & 0.00 & 0.00 & 0.26 & 0.08 & 0.49 & 0.17 \\ Bs & 0.00 & 0.00 & 0.40 & 0.23 & 0.34 & 0.03 \\ S & 0.01 & 0.00 & 0.26 & 0.24 & 0.23 & 0.26 \\ D & 0.00 & 0.00 & 0.37 & 0.17 & 0.43 & 0.03 \end{array}\right) \quad (3)$$

$$P_3 = \left(\begin{array}{c|cccccc} & R & F & Be & Bs & S & D \\ \hline R & 0.00 & 0.00 & 0.00 & 0.00 & 1.00 & 0.00 \\ F & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ Be & 0.00 & 0.00 & 0.24 & 0.39 & 0.31 & 0.07 \\ Bs & 0.02 & 0.00 & 0.57 & 0.09 & 0.25 & 0.07 \\ S & 0.00 & 0.00 & 0.38 & 0.21 & 0.17 & 0.23 \\ D & 0.00 & 0.00 & 0.53 & 0.05 & 0.42 & 0.00 \end{array}\right) \quad (4)$$

These three matrices characterize the transitions in each third of the session and are used to measure differences across the design session. If we examine the transition from F to Be in these parts of the session we can observe a rapid diminution of this activity: in the first third its probability is 0.67, in the second it is 0.00 and also 0.00 in the final third. Similarly, we can trace the behavior of other transitions from the beginning, through the middle, to the end of the design session and use these results to build a quantitatively-based model of designing behavior.

The mean first passage time M is the average number of events traversed before reaching a state from other states. The mean passage time can be obtained from the probability matrix. Kemeny and Snell [10] proved that the mean first passage matrix M is given by:

$$M = (I - Z + EZ_{dg})D \quad (5)$$

Where I is an identity matrix, E is a matrix with all entries of 1, D is the diagonal matrix with diagonal elements $d_{ii} = 1/\alpha_i$, and Z is the fundamental matrix such that

$$Z = (I - (P - A))^{-1} \quad (6)$$

and Z_{dg} is the diagonal matrix of Z. A is the probability matrix, which each row consists of the same probability vector $a = (\alpha_1, \alpha_2, ..., \alpha_n)$ such that $\alpha P = \alpha$.

Equation 7 describes the number of FBS states expected to be passed through from one state before reaching other FBS states.

$$M = \left(\begin{array}{c|cccccc} & R & F & Be & Bs & S & D \\ \hline R & 49.1 & 168.4 & 3.9 & 6.7 & 2.5 & 5.0 \\ F & 70.0 & 168.4 & 1.9 & 6.2 & 3.7 & 4.5 \\ Be & 69.3 & 167.3 & 3.2 & 5.0 & 2.7 & 5.3 \\ Bs & 68.3 & 168.2 & 2.5 & 5.4 & 2.9 & 5.8 \\ S & 67.6 & 166.7 & 3.1 & 5.2 & 3.2 & 4.6 \\ D & 68.6 & 168.1 & 2.7 & 5.4 & 2.6 & 6.0 \end{array}\right) \quad (7)$$

Equations 8, 9 and 10 are the un-normalized mean first passage times matrices of the segments from 1 to 214, 215 to 427 and 428 to 640 respectively. These provide the quantitative bases for an understanding of the character and extent of the iterative nature of the design session through changes in the mean length of the iteration cycle across the session. For example, the mean length of the passage from R to R in the first third is 20.1 and then increases to around 190 in the remainder of the session and the mean length of the iteration cycle of S to Bs starts at 6.9 in the first third, shortens to 5.6 in the

middle third and drops to 4.0 in the final third. These results can then be used to characterize quantitatively this particular design session.

$$M_1 = \left(\begin{array}{c|cccccc} & R & F & Be & Bs & S & D \\ \hline R & 20.1 & 61.6 & 4.4 & 8.3 & 2.8 & 3.6 \\ F & 30.4 & 61.3 & 2.1 & 7.7 & 3.6 & 3.1 \\ Be & 29.6 & 60.0 & 4.1 & 6.7 & 2.6 & 3.2 \\ Bs & 28.8 & 61.3 & 3.1 & 6.8 & 2.9 & 3.8 \\ S & 27.9 & 59.6 & 3.8 & 6.9 & 3.3 & 2.8 \\ D & 29.1 & 61.1 & 3.2 & 6.7 & 2.6 & 4.2 \end{array}\right) \quad (8)$$

$$M_2 = \left(\begin{array}{c|cccccc} & R & F & Be & Bs & S & D \\ \hline R & 192.8 & \infty & 4.3 & 6.6 & 1.0 & 6.4 \\ F & 194.2 & \infty & 4.2 & 6.9 & 3.4 & 6.8 \\ Be & 193.9 & \infty & 3.3 & 6.4 & 2.1 & 5.6 \\ Bs & 194.3 & \infty & 2.9 & 5.6 & 2.5 & 6.6 \\ S & 191.8 & \infty & 3.3 & 5.6 & 2.8 & 5.4 \\ D & 194.1 & \infty & 3.0 & 6.0 & 2.3 & 6.5 \end{array}\right) \quad (9)$$

$$M_3 = \left(\begin{array}{c|cccccc} & R & F & Be & Bs & S & D \\ \hline R & 188.8 & \infty & 3.3 & 5.0 & 1.0 & 8.8 \\ F & 187.7 & \infty & 3.3 & 4.8 & 4.4 & 9.8 \\ Be & 187.1 & \infty & 2.6 & 3.3 & 3.3 & 9.1 \\ Bs & 183.8 & \infty & 2.0 & 4.3 & 3.4 & 9.2 \\ S & 187.8 & \infty & 2.3 & 4.0 & 3.7 & 7.8 \\ D & 188.2 & \infty & 2.1 & 4.4 & 2.9 & 9.6 \end{array}\right) \quad (10)$$

If we assume that each segment is cognitively related to its immediately preceding segment and then select the probabilities of those transitions that represent the eight FBS design processes in Figure 1, we can plot the design processes of this design session for the beginning, middle and end of the session, Figure 5.

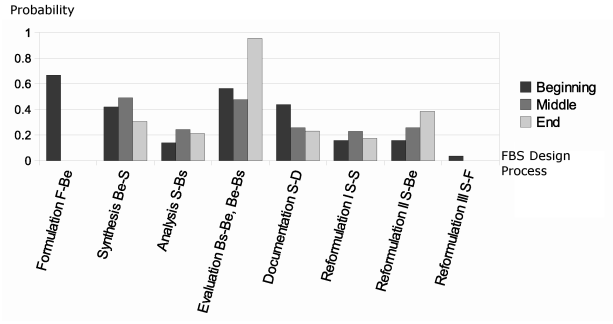

Fig. 5. The probability distributions for the eight FBS design processes for the beginning, middle and end of the session

From the probabilities in Figure 5 we can infer design process behavior for this design session. Formulation only occurs at the beginning of the session. Synthesis increases at the middle of the session and then drops off. Analysis is low at the beginning. There is a significant increase in evaluation and type II reformulation at the end of the session. Again this provides the basis for further qualitative-based characterizations of this design session.

Using different assumptions we can obtain additional results in other dimensions. If we assume that FBS processes occur not with consecutive segments but with related segments we can produce other data for design process behavior. For example when there is a F segment, in order the have a formulation process (F $\rightarrow$ Be), it needs a connection to a later Be segment. With the mean first passage time we can predict when the next Be will appear. With equations 8, 9 and 10 we select those average first passage times that form the eight FBS process shown in Figure 6.

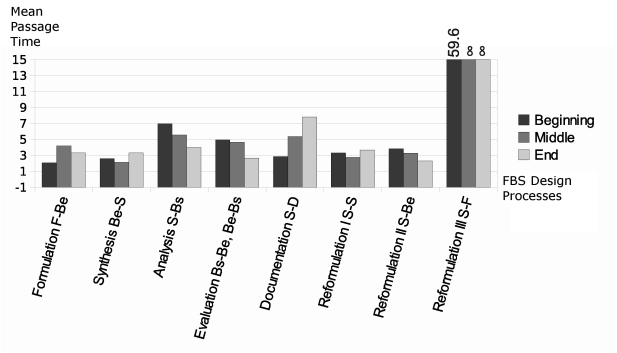

Fig. 6. The distribution of the mean first passage time of the eight FBS design processes

Figure 6 shows there is a change in the behaviors of analysis and documentation when an S design issue occurs. At the beginning of the session there is an average of 2.8 steps for a documentation process that increases to 7.8 steps at the end of the session. Whereas the time expectation of an analysis after an S design issue decreases from 5.2 steps to 4.0 steps.

C. Comparison with Another Domain of Designing

We can compare this software design session with an architectural design session where the task was to generate a conceptual design of a university student union's gallery. The session lasted for 30 minutes. The first 10 minutes was coded for this comparison. Figure 7 compares the code distributions of the first one third of the two design sessions.

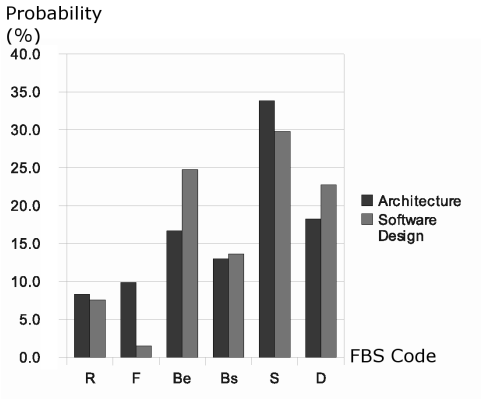

Fig. 7. The FBS code distributions of software design and the architectural design sessions

Since the same generic coding scheme was used for both domains the results are commensurable. Figure 7 shows the architectural design session had considerably more F design issues, less Be design issues, and more S design issues.

These are interesting observations: it says that in the architectural design session, the designers focused more on function, while in the software design session, the designers spent very little time on function-based discussion. A qualitative analysis of the videos of the sessions confirms these quantitative-based observations qualitatively. It is known that in architectural design the same structure can lend itself to a variety of function interpretations. Therefore, in architectural design, the relationship between the set of functions being considered (i.e the main set of purposes for which the structure is being designed) is an open one, as architects continue to create or discover new functions, and continue to reformulate and redefine already identified ones. In the software design session however, the function space appears much more defined and fully formulated from the start. There appear to be a very small number of main, explicitly defined functions, and the designers' reasoning focuses more on expected behavior, derived behavior and structure.

We are now in the position to compare a wide range of measurements of the two domains.

VI. Conclusion

The results in this study demonstrate that the FBS coding scheme can be used to code and quantify the cognitive behavior of software designers. A wide range of measures of cognitive behavior has been shown to be derivable from the basic design issues that are the consequence of the segmentation/coding. Distributions of design issues across the entire design session and any fraction of the session can be measured. When measured across fractions of the session the resulting differences can be tested for statistical significance. Distributions of design processes across the entire design session and any fraction of the session can be measured. When measured across fractions of the session the resulting differences can be tested for statistical significance. Markov models and mean passage time models can be developed for the session and used to develop an understanding of the design cognition of software designers.

Further, since we are using a domain independent coding scheme we can develop comparisons with the design cognition of designers in other domains. In the cases presented in this paper, the software designers were more concerned with behavior compared to the architects who had a greater concern for function and structure (form).

This discussion shows how the FBS ontological framework provides us with a powerful common ground from which to analyze designing activity in various domains. It helps us to develop rich insights into design cognition of software designers to compare the similarities and the differences in design content, knowledge and processes in different domains through an analysis of in-vivo and in-vitro design sessions.

This ontologically-based FBS coding scheme has been applied to statistically significant cohorts of designers and shown to be robust [11].This paper has demonstrated its utility in developing data from design protocols, data that forms the basis of an understanding of the design cognition of software designers.

Acknowledgment

This research is supported in part by a grant from the US National Science Foundation grant no. IIS-1002079. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. We would like to thank Somwrita Sarkar for assistance with the segmentation/coding.

Authors

Jeff Kan is the deputy dean of the School of Architecture, Building and Design, Taylor's University, Malaysia. He completed his PhD in design computing and cognition at the University of Sydney. His study focused on developing and using quantitative methods to study the cognitive behavior of designers.

John Gero is a Research Professor in the Krasnow Institite for Advanced Study. His research fields cover design cognition - understanding the cognition of designing; design computing - using computational constructs to explore design processes; and designing as social behavior - using computational social science to study social interactions during designing. His research has been funded by NSF and DARPA.

References

[1] N. Cross, H. Christiaans, and K. Dorst, *Introduction: the Delft protocols workshop*. John Wily & Son, 1996, pp. 1–14.

[2] J. McDonnell and P. Lloyd, Eds., *DTRS7 Design Meeting Protocols: workshop proceedings*, 2007.

[3] T. R. Gruber, "A translation approach to portable ontology specifications," *Knowledge Acquisition*, vol. 5, no. 2, pp. 199–220, 1993.

[4] J. S. Gero, "Design prototypes: a knowledge representation schema for design," *AI Magazine*, vol. 11, no. 4, pp. 26–36, 1990.

[5] W. T. Kan, "Quantitative methods for studying design protocols," Ph.D. dissertation, The University of Sydney, Faculty of Architecture, Design & Planning, Key Centre of Design Computing and Cognition, 2008.

[6] J. W. T. Kan and J. S. Gero, "A generic tool to study human design activity," in *Human Behavior in Design*, M. Norell Bergendahl, M. Grimheden, L. Leifer, P. Skogstad, and U. Lindemann, Eds., 2009, pp. 123–134.

[7] H. Kenner and J. O'Rourke, "A travesty generator for micros: Nonsense imitation can be disconcertingly recognizable," *BYTE*, no. 12, pp. 449–469, 1984.

[8] A. N. Langville and C. D. Meyer, "Updating markov chains with an eye on google's pagerank," *SIAM Journal on Matrix Analysis and Applications*, vol. 27, no. 4, pp. 968 – 987, 2006.

[9] M. Farbood and B. Schoner, "Analysis and synthesis of palestrina-style counterpoint using markov chains," in *Proceedings of International Computer Music Conference*, Havana, Cuba, 2001.

[10] J. G. Kemeny and J. L. Snell, *Finite Markov Chains*, ser. The University Series in Undergraduate Mathematics. D. Van Nostrand Company, Inc. Princeton, New Jersey, 1960.

[11] H.-H. Tang, Y. Y. Lee, and J. S. Gero, "Comparing collaborative co-located and distributed design processes in digital and traditional sketching environments: A protocol study using the functionbehaviorstructure coding scheme," *Design Studies*, vol. 32, no. 1, pp. 1–29, 2011.